Spring 2017

A Rigorous Take on Randomness: Defining 1-Randomness using Complexity, Measure Theory, and Martingales

Neil A. Dokurno
dokurnon@gmail.com

Abstract

By flipping a coin repeatedly and recording the result, we can create a sequence that intuitively is random. This paper presents a formal definition of 1-randomness widely (but not universally) thought to capture this informal notion. The attempt builds on the success and spirit of work from the 20th century that formalized the notion of computability with Turing machines. We will begin by presenting some of the characteristics of Turing machines. With that background, we will first use universal prefix-free Kolmogorov complexity to characterize randomness. The exploration of complexity will involve a thorough proof of the KC Theorem and an introduction to information content measures. Next, we will approach randomness using measure theory together with Σ_1^0 classes. And third, we will characterize randomness using left computably enumerable functions called martingales and supermartingales. Finally, we will prove that each of these characterizations is equivalent; i.e., the same sequences are random according to each of them.

All nature is but art, unknown to thee;
All chance, direction, which thou canst not see;
All discord, harmony not understood;
All partial evil, universal good.

"An Essay on Man" by Alexander Pope

1 Computability Theory

The understanding of randomness we expound below builds on results from computability theory, which itself arose out of formal logic, but we need not go into that here.[1] We will only explore enough to understand our approaches to randomness. But before computability can be discussed, we need some grasp on Turing machines. And to understand Turing machines, the following vocabulary is needed.

Definition 1.1. 1. A Turing machine has a *tape* that is subdivided into an infinite number of *cells*. Intuitively, tapes are like an ideal hard drive of infinite size where each cell stores one and only one bit of information.

2. Each cell has one *tape symbol* out of infinite possible states denoted $S_0, S_1, S_2, \ldots$. Generally, we only use S_0 and S_1, which we denote 0 and 1 respectively. The symbol 0 intuitively means "blank." A tape symbol is the one bit of information each cell stores.

3. A *reading head* reads the tape symbol of one particular cell at a time.

4. The reading head has one *internal state* out of an infinite possibility of states denoted $q_0, q_1, \ldots$. We choose an initial state for the reading head.

5. There are four *action symbols* that tell the reading head what to do:

   - L: reading head moves one cell left.
   - R: reading head moves one cell right.
   - 0: the current cell is given the tape symbol 0. Intuitively, this erases the cell.
   - 1: the current cell is given the tape symbol 1.

6. A *quadruple* $Q = q_iSAq_j$ says, "if the reading head is in state q_i and the current cell has symbol S, then perform action A and change the internal state to q_j."

7. A set X of quadruples is said to be *consistent* if

$$q_iSAq_j, q_iSA'q_k \in X \Rightarrow A = A' \text{ and } j = k.$$

   In other words, reading heads in the same location and in the same state will be told to act in at most one way.

Definition 1.2. 1. A *Turing machine* M is a finite and consistent set of quadruples.

2. To input $n \in \mathbb{N}$ to M, we put the "1" symbol in $n+1$ consecutive cells and the "0" symbol in all other cells. We place the reading head at the leftmost cell with a "1" symbol.

[1] Barry S. Cooper's *Computability Theory* provides a more in depth presentation of computability theory.

3. For multiple inputs, we separate the inputs by a cell with symbol "0" and nevertheless start with the reading head at the leftmost cell with a 1.

4. When no quadruple in M applies, the machine *halts* and outputs the total number of cells with the symbol "1". If a machine M with input n outputs m, we write $M(n) = m$.

Note that a Turing machine is not an ordered list of instructions like many computer programs. Nevertheless, it can be shown that all computer programs could be replicated using only Turing machines. The relation to computer programs provides evidence for the Turing thesis, which will be introduced shortly.

Observe that consistency guarantees that if a machine halts on a specific input, then that output will be unique. Thus, we can think of Turing machines as functions.

Definition 1.3. A *partial function* $f : \mathbb{N} \to \mathbb{N}$ is a function $f : A \to \mathbb{N}$ for some $A \subseteq \mathbb{N}$. For $n \in A$, we write $f(n) \downarrow$ to denote that $f(n)$ is defined. In this case, we say $f(n)$ *halts*. Otherwise for $n \notin A$, we write $f(n) \uparrow$ and say $f(n)$ does not halt.

When $A = \mathbb{N}$, we say that f is *total*.

The following definition justifies our usage of "halt." It will also be the central definition in our study of computability.

Definition 1.4. A partial function f is said to be *Turing computable* if there exists a Turing machine M such that $f(n) \downarrow$ implies $f(n) = M(n)$ and $f(n) \uparrow$ implies $M(n)$ does not halt.

Below we provide a simple example of a Turing computable total function. When writing even a straightforward Turing machine, detailed comments are necessary for the reader.

Example 1.5. We will show that $f(x) = \begin{cases} 0 & x = 0, \\ 1 & x \neq 0 \end{cases}$ is Turing computable.

We can compute f with the Turing machine given as follows:

$q_0 1 0 q_1$	delete the first 1 on the tape (since we input $n+1$)
$q_1 0 R q_2$	move reading head one cell right
$q_2 1 R q_3$	move reading head right if current cell has symbol "1";
	note machine halts with output 0 if cell symbol is "0"
$q_3 1 0 q_4$ $q_4 0 1 q_3$	delete any remaining "1" symbols on the tape so that the machine outputs 1.

While it would be difficult to figure out an undocumented Turing machine, even this example illustrates the connection to algorithms. In fact, Alan Turing claimed that these notions are interchangeable. However, we cannot state his claim as a theorem because the notion of an algorithm is informal. Moreover, Turing computability is a way to formalize the notion of an algorithm.

Remark 1.6.

1. We say a partial function f is *partial computable* or p.c. if intuitively there is an algorithm or effective procedure for computing its outputs.

2. We say a partial function f is *computable* if f is both total and partial computable.

The Turing Thesis. A partial function f is partial computable if and only if it is Turing computable.[2]

[2]In the literature, the Turing thesis is usually combined with a similar thesis by Church to create the landmark Church-Turing thesis. The Church thesis identifies the p.c. functions with the *partial recursive* functions. In fact, all computability results within this paper could be restated using recursive functions. However, we will only be using Turing machines, so we will need only the Turing thesis.

The Turing thesis is a cornerstone of computability theory. It allows us to bypass the intricacies of building a Turing machine and instead use intuitive algorithms. In this brief introduction we cannot argue justly for accepting this thesis, but arguments can be readily found in Cooper [**?**].

We are often interested in talking about sets. We have two relevant notions.

Definition 1.7. 1. A set $A \subseteq \mathbb{N}$ is *computably enumerable* or c.e. if either $A = \varnothing$ or there is a p.c. function f such that $A = \operatorname{im} f$.

2. A set $A \subseteq \mathbb{N}$ is *computable* if both $\overline{A}$ and A are computably enumerable.

3. A n-ary relation A is *computable* if the set $\{(x_1, \ldots, x_n) \in A\}$ is computable.

Remark 1.8. 1. By the Turing thesis, a set A is computably enumerable if and only if there is an effective algorithm for listing all members of A. Note that such an algorithm need not finish in a finite amount of time.

2. Similar to the first comment, a set $A \subseteq \mathbb{N}$ is c.e. if for all $x \in \mathbb{N}$ there is a effective procedure that halts when $x \in A$. There need not be a procedure that halts for all $x \notin A$.

3. Likewise, a set A is computable if and only if for all $n \in \mathbb{N}$ there is an effective procedure for determining whether $n \in A$ or $n \notin A$. Sets with this property are said to be *decidable*.

4. From the Turing thesis viewpoint, it is clear that all computable sets are c.e., but not all c.e. sets are computable.

Using the Turing thesis, we can provide an easy but important example of a computable set.

Example 1.9. The set $\mathbb{N}$ itself is computable. We can enumerate $\mathbb{N}$ by the following procedure.
Step 0: enumerate 0 into $\mathbb{N}$.
Step n: enumerate n into $\mathbb{N}$.

When we enumerate a set using the Turing thesis, we often provide procedures in this form. Now, $\overline{\mathbb{N}} = \varnothing$, which is c.e. Thus, $\mathbb{N}$ is computable.

We also could have defined a c.e. set using the domain of a Turing computable function.

Proposition 1.10. *A is c.e. if and only if $A = \operatorname{dom} f$ for some some Turing computable function f.*

Proof. [**?**] ($\Rightarrow$) This proof will further demonstrate how the Turing thesis can be used. Suppose A is c.e. Then $A = \operatorname{im} f$ for some Turing computable function f. We can define a function g with $\operatorname{dom} A$ as follows.
Step 0: Using its effective procedure, compute $f(0)$ for 1 second. If $f(0) \downarrow = x$, define $g(x) = 0$. Otherwise, proceed to the next step.
Step n: Compute $f(m)$ for $n + 1$ seconds for all $m \leq n + 1$. If some $f(m) \downarrow = y$ and $g(y)$ is not already defined, then define $g(y) = m$. It is effective to check if y is already in the domain of g since during step n, the domain of g is finite. Any procedure that involves only finitely many steps eventually halts.

Notice $\operatorname{dom} g = \operatorname{im} f = A$, and g is partial computable since it is defined effectively. Hence, there is a p.c. function g such that $A = \operatorname{dom} g$.

($\Leftarrow$) Suppose $A = \operatorname{dom} g$ for some partial computable function g. Let M be the machine that computes g. The following will be an effective procure to enumerate A.
Step 0: Run M on input 0 through at most 1 stage; i.e., let M satisfy at most 1 quadruple and

then manually halt the machine. If M halts by itself, enumerate 0 into A.
Step n: Run M on each input $k \leq n$ through $n+1$ stages. Whenever M manually halts on k, enumerate k into A.

Eventually, all members of A will be enumerated since $x \in A \Leftrightarrow f(x) \downarrow \Leftrightarrow M(a)$ halts. Thus, $A = \operatorname{dom} g$ is c.e. $\dashv$

Remark 1.11. Notice that our enumeration A is not necessarily efficient in that the same element of A gets enumerated many times. This does not matter since computation time is not a concern for us here.

In the algorithm, we specified a length of time to run a machine. We also specified the number of stages or quadruples to run. These are useful tools for forcing a machine to halt, so we introduce the following notation.

Definition 1.12. For a Turing machine M, we write $M[s]$ to denote running M either until it halts or until s seconds elapse. Likewise, we write $f[s]$ to denote running the machine that computes f for s seconds.

We would like to extend our definitions to collections of objects.

Definition 1.13.
1. A collection of functions $\{f_n\}_{n\in\mathbb{N}}$ is said to be *uniformly partial computable* or uniformly p.c. if there is a partial computable function $f : \mathbb{N}^2 \to \mathbb{N}$ defined by $f(n, x) = f_n(x)$.
2. A collection of function $\{A_n\}_{n\in\mathbb{N}}$ is said to be *uniformly computably enumerable* or uniformly c.e. if there is a partial computable collection of functions $\{f_n\}_{n\in\mathbb{N}}$ such that $\operatorname{im} f_n = A_n$ for each $n \in \mathbb{N}$.

Remark 1.14. By the Turing thesis, showing that a collection of functions $\{f_n\}_{n\in\mathbb{N}}$ is uniformly p.c. is equivalent to building an algorithm that takes two inputs n and x such that when n is fixed, the algorithm computes f_n.

At this point, we might ask whether the set of all partial computable functions is c.e. The answer is yes. We will effectively encode each Turing program into a unique number. The idea is that we can also effectively decode the number to get the program.

Definition 1.15. Let p_n denote the nth prime number. The *Gödel number gn* of a collection of symbols is defined recursively as follows:

- $\operatorname{gn}(L) = 2$
- $\operatorname{gn}(R) = 3$
- $\operatorname{gn}(q_i) = p_{2+2i}$
- $\operatorname{gn}(S_i) = p_{2+2i+1}$
- For a quadruple Q, $\operatorname{gn}(Q) = 2^{\operatorname{gn}(q_i)}3^{\operatorname{gn}(S)}5^{\operatorname{gn}(A)}7^{\operatorname{gn}(q_j)}$
- For a program $P = \{Q_0, Q_1 \ldots, Q_k\}$, $\operatorname{gn}(P) = 2^{\operatorname{gn}(Q_0)}3^{\operatorname{gn}(Q_1)} \ldots p_{k+1}^{\operatorname{gn}(Q_k)}$ where p_{k+1} is the $(k+1)$th prime number.

Remark 1.16. We set $\text{gn}(0) = 3$ and $\text{gn}(1) = 5$ whether these symbols are used as tape or action symbols. The dual use of 0 and 1 might seem problematic for the uniqueness of a program's Gödel number. However, for $Q = q_0 10 q_1$ and $Q' = q_0 01 q_1$, note $\text{gn}(Q) \neq \text{gn}(Q')$ since $2^3 3^5 \neq 2^5 3^5$.

By the uniqueness of prime factorization, each program receives a unique Gödel number. And the encoding and decoding are intuitively effective. Thus, the set of all p.c. functions is c.e. For a rigorous account that can be adapted to our situation, the reader can consult sections 3.3 and 3.4 of Enderton [**?**].

Definition 1.17. 1. We define

$$P_e = \begin{cases} P & \text{if } gn^{-1}(e) \downarrow \text{ and equals some Turing program } P, \text{ and} \\ \varnothing \text{ (the empty program)} & \text{otherwise.} \end{cases}$$

We call e the *index* of P.

2. We define $\varphi_e^{(k)}$ by the k-place partial function computed by P_e.

Using the uniqueness of e for each program, we can define a machine that computes all Turing machines.

Proposition 1.18. There exists a *universal Turing machine* $\mathcal{U}$ which computes the function $\Phi : \mathbb{N}^2 \to \mathbb{N}$ such that $\Phi(e, x) = \varphi_e(x)$ for all $e, x \in \mathbb{N}$.

Proof. [**?**] We will use the Turing thesis to show that Φ as defined above is Turing computable. Take any $e, x \in \mathbb{N}$. Given how programs are encoded, we can effectively decode e by finding its prime factorization and the prime factorization of the exponent of each prime number. Once e has been decoded, we can effectively compute $\varphi_e(x)$. Thus, Φ is Turing computable by the Turing thesis. Let $\mathcal{U}$ be the machine that computes Φ. $\dashv$

Sometimes, we will be given a two place p.c. function, but we want to compute it using one-place functions. The following lemma allows us to do that.

Lemma 1.19. (s-m-n Theorem) For $g : \mathbb{N}^2 \to \mathbb{N}$ where $g(x, y) = \varphi_e^{(2)}(x, y)$ for some e, there exists a partial computable $s : \mathbb{N} \to \mathbb{N}$ such that $g(x, y) = \varphi_{s(x)}^{(1)}(y)$.

Proof. [**?**] For a fixed $x \in \mathbb{N}$, let N_x be the Turing machine that takes y and inputs (x, y) to M where M computes f. Then let $s(x)$ encode N_x. We know $s(x)$ is p.c. since the sets N_x can be enumerated. It follows that $g(x, y) = \varphi_{s(x)}^{(1)}(y)$. $\dashv$

Theorem 1.20. (Recursion Theorem) If f is a total computable function, then there exists a $k \in \mathbb{N}$ such that $\varphi_{f(k)} = \varphi_k$. We call k a *fixed-point* of f.

Proof. [**?**] Using the s-m-n theorem, find an $s(x)$ such that $\varphi_{s(x)} = \varphi_{\varphi_x(x)}$. We claim the set of partial computable functions is closed under composition. Given p.c. functions f and g, we can compute $g \circ f$ with the following algorithm. Take any $n \in \mathbb{N}$. First, compute $f(n)$ using the effective algorithm f. Then compute $g(f(n))$ using the effective algorithm for g. Consequently, the composition $g \circ f$ is p.c. Thus, $f \circ s$ is p.c. and has an index e. So $\varphi_{f(s(e))} = \varphi_{\varphi_e(e)} = \varphi_{s(e)}$. Therefore, taking $k = s(e)$ gives the desired result. $\dashv$

Heretofore, we have looked at computability of functions defined on the natural numbers $\mathbb{N}$. In what follows, we are interested in sequences of ones and zeros.

Definition 1.21. A *string* is a finite sequence $\sigma = x_0 \dots x_n$ where $x_i \in \{0, 1\}$ for $0 \leq i \leq n$. The *length* of σ is $n + 1$, which we write $|\sigma| = n + 1$. For two strings $\sigma = x_0 \dots x_m$ and $\tau = y_1 \dots y_n$, the *concatenation* of σ with τ is $\sigma\tau = x_0 \dots x_n y_0 \dots x_m$.

Remark 1.22.

1. We denote the space of all strings by $2^{<\omega}$.
2. By treating strings as the binary expansion of natural numbers, there is clearly a bijection between $2^{<\omega}$ and $\mathbb{N}$.
3. Consequently, we will interchangeable denote the natural numbers by ω and $\mathbb{N}$. Usually, but not always, ω will be used to emphasis sequences.
4. We denote the space of all infinite sequences of ones and zeros by 2^{ω}.
5. We usually use lowercase Greek letters to name strings and uppercase Roman letters to denote infinite sequences of ones and zeros. The letter λ denotes the empty sequence.

We can interpret our inputs and outputs of Turing machines as strings since it is intuitively effective to convert from binary to base_{10} and vice versa. From this viewpoint, the function defined by $\sigma \mapsto |\sigma|$ is Turing computable by the empty machine that just outputs its input.

We will need only a bit more notation before we get started on randomness.

Definition 1.23. For a sequence $X = x_0 x_1 \dots$ in 2^{ω}, we write $X(n) = x_n$. We define $\sigma(n)$ analogously for $n < |\sigma|$. For $\sigma, \tau \in 2^{<\omega}$ with $|\sigma| < |\tau|$, we write $\sigma \preceq \tau$ if either $\sigma = \tau$ or $\sigma(n) = \tau(n)$ for all $n < |\sigma|$. If neither $\tau \preceq \sigma$ nor $\sigma \preceq \tau$, then we say that σ and τ are *incomparable*, which we denote $\sigma \mid \tau$. Likewise, for $X \in 2^{\omega}$ we write $\sigma \prec X$ if $\sigma(n) = X(n)$ for all $n < |\sigma|$.

For $\sigma \in 2^{<\omega}$, we say the *set of extensions of* σ is $[\sigma] = \{X : \sigma \prec X\}$.

With this background in computability theory, we can proceed to randomness.

2 Approaching Randomness with Kolmogorov Complexity

Our first approach to algorithmic randomness will make direct use Turing machines. To recall, Turing machines take an input, manipulate it according to some set of instructions, and then produce an output. From here on, we will be interpreting the input and output of Turing machines as finite strings. The main idea of this machine-centric approach to randomness is that no algorithm should be able to output a random sequence using less information than contained in the whole. In a sense, random sequences cannot be compressed. It will require substantial work before we can consider randomness. First, we will consider plain Kolmogorov complexity.

Definition 2.1. The *plain Kolmogorov Complexity* of a string $\sigma \in 2^{<\omega}$ with respect to a partial computable function $f : 2^{<\omega} \to 2^{<\omega}$ is

$$C_f(\sigma) = \min\{|\tau| : f(\tau) = \sigma\}.$$

When $f(\tau) \neq \sigma$ for all $\tau \in 2^{<\omega}$, we say that $C_f(\sigma) = \infty$.

The plain Kolmogorov complexity of a string σ relative to a machine M is the string τ of the least length such that $M(\tau) = \sigma$. The following example demonstrates the importance of our choice of M.

Example 2.2. 1. Define f to the constant function, which is computable by the empty program. Then $C_f(\sigma) = |\sigma|$ for all $\sigma \in 2^{<\omega}$.

2. Define f' by the machine M' that for input τ outputs the string with $|\tau|$ zeros. Then if $\sigma(n) = 1$ for any $n < |\sigma|$, we have $C_{f'}(\sigma) = \infty$.

In order to be independent of the choice of f, we will focus on $C_{\mathcal{U}}$ for the Universal Turing machine $\mathcal{U}$ defined in proposition 1.18. There we assigned to each p.c. function f a Gödel number, which in binary is a coding sequence $\rho_f \in 2^{<\omega}$. In fact, there are other universal machines that use different encodings. We will be general enough that the choice of $\mathcal{U}$ will not be relevant. The following result establishes the minimality up to a constant of complexity relative to a universal machine.

Proposition 2.3. We have $C_{\mathcal{U}}(\sigma) \leq C_f(\sigma) + |\rho_f|$ for all $\sigma \in 2^{<\omega}$ and all partial computable functions f.

Proof. Let $\sigma \in 2^{<\omega}$ be arbitrary. Take the partial computable function f with coding sequence ρ_f. Then

$$\begin{aligned} C_{\mathcal{U}}(\sigma) &= \min\{|\tau| : \mathcal{U}(\tau) = \sigma\} \\ &\leq \min\{|\rho_f\tau| : \mathcal{U}(\rho_f\tau) = \sigma\} \\ &= \min\{|\rho_f\tau| : f(\tau) = \sigma\} \\ &\leq \min\{|\tau| + |\rho_f| : f(\tau) = \sigma\} \\ &= \min\{|\tau| : f(\tau) = \sigma\} + |\rho_f| \\ &= C_f(\sigma) + |\rho_f|. \end{aligned}$$

Note that this result holds for any method of encoding the Turing machines. Thus, for two universal machines $\mathcal{U}, \mathcal{U}'$, we have $C_{\mathcal{U}} = C_{\mathcal{U}'} + c$ for some constant $c \in \mathbb{N}$. It thus makes sense to set $C = C_{\mathcal{U}}$ and call it the plain Kolmogorov complexity. $\dashv$

The plain Kolmogorov complexity of a string σ can be interpreted as how much information is needed to describe σ. Here, a description of σ just amounts to being able to identify $\sigma(n)$ for each $n < |\sigma|$, and a description of σ requires more information than a description of σ' if $C(\sigma) > C(\sigma')$. However, there is a problem with using plain Kolmogorov complexity for this goal in that C has access to more information than that only contained in σ. In fact, a machine M on input τ can also use $|\tau|$ in order to compute $M(\tau)$. Our second example above is such a machine. So we have cases where $C(\sigma) = C(\sigma')$ yet $C(\sigma)$ uses this additional information and $C(\sigma')$ does not. Consequently, the equality ignores a difference of needed information.

We consider the prefix-free Kolmogorov complexity in order to avoid this ambiguity.

Definition 2.4. A set $A \subseteq 2^{<\omega}$ is *prefix-free* if for all $\sigma, \tau \in A$ we have

$$\sigma \preceq \tau \Rightarrow \sigma = \tau.$$

A partial computable function f is said to be *prefix-free* if the its domain is prefix-free. Likewise, a Turing machine M is said to be prefix-free if its domain is prefix-free.

Downey and Hirschfeldt note the phone numbers are an example of a prefix-free set since no phone number is a proper prefix of another [**?**]. For example, there is no valid phone number in the U.S. that looks like 911-XXX-XXXX since 911 is already a phone number. Consequently, we do not need a special symbol to mark the last digit of someone's number. This simple idea of a prefix-free set will be a powerful tool going forward.

Again, we can talk about the Kolmogorov complexity of a finite string but in relation to a prefix-free function. We will write K_f rather than C_f if f is prefix-free. As before, we would like to consider a universal prefix-free machine. The next theorem establishes that a prefix-free Universal machine $\mathcal{U}$ for prefix-free functions exists. In general, proofs involving prefix-free sets will be more intricate than their plain counterparts. Here, we must establish that the prefix-free sets can be indexed and the resulting universal function is also prefix-free.

Proposition 2.5. There is a universal prefix-free Turing machine.

Proof. [**?**] First, we will define an enumeration of all and only the prefix-free partial computable functions. For any $e \in \omega$, define $\psi_e[s]$ by $\varphi_e[s]$ if $\operatorname{dom}\varphi_e[s]$ is prefix-free and $\psi_e[s-1]$ otherwise.

We claim that $\psi_e[s]$ is prefix-free for each $s \in \mathbb{N}$. Let's show this by induction on s. For $s = 1$, if $\varphi_e[1]$ is not prefix-free, then $\operatorname{dom}(\psi_e[1]) = \operatorname{dom}(\psi_e[0]) = \varnothing$, which is prefix-free. Otherwise, $\varphi_e[1]$ is prefix-free and $\psi_e = \varphi_e$. Now, suppose $\psi_e[t]$ is prefix free for some $t \in \mathbb{N}$. If $\varphi_e[t+1]$ is prefix-free, then $\psi_e[t+1] = \varphi_e[t+1]$ is prefix-free. Otherwise, $\psi_e[t+1] = \psi_e[t]$, which is prefix-free by our induction hypothesis. Thus, $\psi_e[s]$ is prefix-free for all $s \in \mathbb{N}$.

Now, let $\sigma, \tau \in \operatorname{dom}\psi_e$ be arbitrary. Then there exists some s_1 and s_2 such that $\sigma \in \operatorname{dom}\psi_e[s_1]$ and $\tau \in \operatorname{dom}\psi_e[s_2]$. Let $s = \max(s_1, s_2)$. Then $\sigma, \tau \in \operatorname{dom}\psi_e[s]$, which is prefix-free. So neither $\tau \preceq \sigma$ nor $\sigma \preceq \tau$. Thus, ψ_e is prefix-free.

And if φ_e is a prefix-free computable function, then $\operatorname{dom}\varphi_e[s]$ is prefix-free for all s. So $\psi_e = \varphi_e$. Consequently, $\{\psi_e\}_{e\in\omega}$ is an enumeration all and only the prefix-free partial computable functions. Using this enumeration, define

$$\Psi(1^e 0\sigma) = \psi_e(\sigma).$$

Since $\{\psi_e\}_{e\in\omega}$ enumerates all partial-computable functions, Ψ is universal.

It only remains to show that Ψ is prefix-free. Let $\sigma \in \operatorname{dom}\Psi$ be arbitrary. Take any string τ with $\sigma \preceq \tau$. Then there is some $n \in \omega$ such that $\sigma(n+1) = \tau(n+1) = 0$ and $\sigma(i) = \tau(i) = 1$ for all $i \leq n$. By definition of Ψ, we have $e = n$. So $\sigma = \rho\sigma'$ and $\tau = \rho\tau'$ where $\rho = 1^n 0$ and $\sigma' \preceq \tau'$. Hence, $\Psi(\sigma) = \psi_n(\sigma')$. Since ψ_n is prefix-free, $\tau' \notin \operatorname{dom}\psi_n$. Hence, $\Psi(\tau) = \psi_n(\tau')$ is undefined, which implies Ψ is prefix-free. The same argument that showed Φ is p.c. also applies for Ψ. Therefore, Ψ is a universal prefix-free partial computable function. The corresponding Turing machine must be a universal prefix-free machine. $\dashv$

In fact, there are many such machines depending on how we encode the prefix-free p.c. functions. We will fix a prefix-free universal Turing machine and denote it by $\mathcal{U}$. We are only interested in prefix-free complexity from here on, so $\mathcal{U}$ will no longer refer to the universal Turing machine of section 1.

The following theorem uses the Recursion Theorem 1.21 of the preceding section to prove a similar result for prefix-free machines.

Theorem 2.6. (Recursion Theorem for Prefix-Free Machines) For any partial computable function $h : 2^{<\omega} \times \mathbb{N} \to 2^{<\omega}$ such that for each e the function $g_e : 2^{<\omega} \to 2^{<\omega}$ defined by $n \mapsto h(n, e)$ is prefix-free, we can compute an index k such that $\psi_k = g_k$.

Proof. Define a function $f : \mathbb{N} \to \mathbb{N}$ using the s-m-n theorem such that f is a total computable function and for all $e \in \mathbb{N}$ we have $\varphi_{f(e)} = g_e$. Then by theorem 1.21 we can compute some $k \in \mathbb{N}$ such that $\varphi_{f(k)} = \varphi_k$. Hence, $\varphi_k = g_k$. Since g_k is prefix-free by assumption, φ_k is prefix-free. Hence, $\varphi_k = \psi_k$ by construction of ψ_k. Thus, $\psi_k = g_k$. $\dashv$

Just as we set $C_{\mathcal{U}} = C$ for a universal machine of all Turing machines, we want to set $K_{\mathcal{U}} = K$ for all prefix-free machines. Again, the concern is that there are many universal prefix-free machines. Nevertheless, the following result justifies this notation. The result will also be useful by itself.

Proposition 2.7. If M is a prefix-free Turing machine, then for all $\sigma \in 2^{<\omega}$ we have

$$K_{\mathcal{U}}(\sigma) \leq K_M(\sigma) + c$$

for some constant $c \in \mathbb{N}$.

Proof. Let M be a prefix-free Turing machine that computes ψ_e. Let $\sigma \in 2^{<\omega}$ be arbitrary. Then

$$\begin{aligned} K_{\mathcal{U}}(\sigma) &= \min\{|\tau| : \mathcal{U}(\tau) = \sigma\} \\ &\leq \min\{|1^e 0\tau| : \mathcal{U}(1^e 0\tau) = \sigma\} \\ &\leq \min\{|1^e 0| + |\tau| : M(\tau) = \sigma\} \\ &= K_M(\sigma) + |1^e 0| = K_M(\sigma) + c \text{ for some constant } c \in \mathbb{N}. \end{aligned}$$

Thus, $K_{\mathcal{U}}$ is minimal as a universal prefix-free machine. Since the result can be generalized to any encoding of a universal prefix-free machine, all universal prefix-free machines differ at most by a constant. This justifies our notation $K = K_{\mathcal{U}}$. $\dashv$

One of our major tools in handling prefix-free Kolmogorov complexity will be the KC theorem. It allows us to construct prefix-free machines. Also, this result suggests a connection between the complexity approach of this section and the measure theory approach of the next.

Lemma 2.8. For all $n \geq 1$, we have $1 - 2^{-n} = \sum_{i=1}^{n} 2^{-n}$.

Proof. We proceed by induction on n. For the base case $n = 1$, note $1 - 2^{-1} = 1 - \frac{1}{2} = \frac{1}{2} = \sum_{i=1}^{1} 2^{-i}$.

For the induction hypothesis, suppose for some $k \geq 1$ that $1 - 2^{-k} = \sum_{i=1}^{k} 2^{-i}$. Then

$$\begin{aligned} 1 - 2^{-k} = \sum_{i=1}^{k} 2^{-i} &\Rightarrow 2^{k+1} - 2 = 2^{k+1} \sum_{i=1}^{k} 2^{-i} = \sum_{i=1}^{k} 2^{(k+1)-i} = -1 + \sum_{i=1}^{k+1} 2^{(k+1)-i} \\ &\Rightarrow 2^{k+1} - 1 = \sum_{i=1}^{k+1} 2^{(k+1)-i} \\ &\Rightarrow 1 - 2^{-(k+1)} = \sum_{i=1}^{k+1} 2^{-i} \text{ by dividing both sides by } 2^{k+1}. \end{aligned}$$

Thus, by induction, $1 - 2^{-n} = \sum_{i=1}^{n} 2^{-n}$ for all $n \in \mathbb{N}$. $\dashv$

Theorem 2.9. (KC Theorem) If a computable sequence of pairs $(d_i, \tau_i)_{i \in \omega}$, which are called *requests*, are such that $d_i \in \mathbb{N}$ and for $\tau_i \in 2^{<\omega}$ we have $\sum_i 2^{-d_i} \leq 1$, then there exists a prefix-free Turing machine M and strings σ_i with $|\sigma_i| = d_i$ such that $M(\sigma_i) = \tau_i$ for all $i \in \omega$ and $\operatorname{dom} M = \{\sigma_i : i \in \omega\}$.

Proof. [**?**] Let $(d_i, \tau_i)_{i \in \omega}$ be such that $d_i \in \mathbb{N}$ and $\tau_i \in 2^{<\omega}$ for each i. Assume $\sum_i 2^{-d_i} \leq 1$.

We will use an indirect device in this proof that can be credited to Joe Miller. For each $n \in \omega$, we define $x^n = x_1^n x_2^n \ldots x_m^n \in 2^{<\omega}$ so that

$$0.x_1^n \ldots x_m^n = 1 - \sum_{j \leq n} 2^{-d_j}.$$

Note $\sum_{j \leq n} 2^{-d_j} \leq \sum_{1 \leq j} 2^{-d_j} \leq 1$ by our assumption, so $0 \leq 0.x_1^n \ldots x_m^n$ for all $n \in \omega$. We understand this decimal in base two. This means that $0.x_1^n \ldots x_m^n = \sum_{x_l^n = 1} 2^{-l}$. For example, $0.1 = 2^{-1} = 1/2$, $0.101 = 2^{-1} + 2^{-3} = 5/8$, and $0.0111 = 2^{-2} + 2^{-3} + 2^{-4} = 11/16$.

For each $n \in \omega$, we aim to effectively construct a string σ_n so that for all l where $x_l^n = 1$ there exists a string μ_l^n of length l so that $S_n = \{\sigma_i : i \leq n\} \cup \{\mu_l^n : x_l^n = 1\}$ is prefix-free. Roughly, the μ_l^n keep track of the strings are incomparable with $\sigma_1, \ldots, \sigma_n$.

For the base case, set $\sigma_0 = 0^{d_0} = \underbrace{0 \ldots 0}_{d_0 \text{ times}}$. Clearly, $|\sigma_0| = d_0$. To see that this choice is effective, note that $0 \mapsto (d_0, \tau_0)$ is computable by assumption. Further, the machine that sends $(a, b) \mapsto a$ is total computable, and the function $n \mapsto 0^n$ is total computable as well. By composing these functions, we get a computable function that computes σ_0. So this choice is effective.

We claim that $x_l^0 = 1 \Leftrightarrow 0 < l \leq d_0$. By definition of $x^0 = x_1^0 \ldots x_m^0$, so we have $0.x_1^0 \ldots x_m^0 = 1 - \sum_{i=0}^{0} 2^{-d_i} = 1 - 2^{-d_0} = \sum_{i=1}^{d_0} 2^{-i}$ by lemma 2.8. In base two, observe that

$$\sum_{i=1}^{d_0} 2^{-i} = 0.\underbrace{11 \ldots 1}_{d_0 \text{ times}}.$$

For example, if $d_0 = 3$, then $\sum_{i=1}^{d_0} 2^{-i} = 1/2 + 1/4 + 1/8 = 0.1 + 0.01 + 0.001 = 0.111$. Consequently, $x_l^0 = 1$ if and only if $1 \leq l \leq d_0$. For each l such that $x_l^0 = 1$, define $\mu_l^0 = 0^{l-1}1$. Note $|0^{l-1}1| = l$. There are only finitely many μ_l^0, so our choice of them is effective.

We claim that $S_0 = \{\sigma_0\} \cup \{\mu_l^0 : x_l^0 = 1\}$ is prefix-free. Let $\mu_l^0, \mu_k^0 \in S_0$ be arbitrary. Then $1 \leq l \leq d_0$ and $1 \leq k \leq d_0$. Without loss of generality, suppose $l \leq k$. As $\mu_l^0 = 0^{l-1}l$ and $\mu_k^0 = 0^{k-1}1$, we have

$$\mu_l^0 \preceq \mu_k^0 \Leftrightarrow l = k \Leftrightarrow \mu_l^0 = \mu_k^0.$$

Also, $\sigma_0(l) = 0 \neq \mu_l^0$, so $\mu_l^0 \not\preceq \sigma_0$. And $\sigma_0 \not\preceq \mu_l^0$ since either $|\sigma_0| > |\mu_l^0|$ or $0 = \sigma_0(l) \neq \mu_l^0)(l) = 1$. Thus, S_0 is a prefix-free set.

For the induction hypothesis, suppose we have effectively constructed both $\sigma_0, \ldots, \sigma_k$ and μ_l^k with $|\sigma_i| = d_i$ for each $i \leq k$ and $|\mu_l^k| = l$ for each $x_l^k = 1$ so that $S_k = \{\sigma_i : i \leq k\} \cup \{\mu_l^k : x_l^k = 1\}$ is prefix-free.

On the one hand, suppose $x_{d_{k+1}}^k = 1$. Then

$$\begin{aligned}
0.x^{k+1} = 1 - \sum_{i \leq k+1} 2^{-d_i} &= 1 - 2^{-d_{k+1}} - \sum_{i \leq k} 2^{-d_i} \\
&= (1 - \sum_{i \leq k} 2^{-d_i}) - 2^{-d_{k+1}} \\
&= 0.x^k - 2^{-d_{k+1}} \\
&= 0.x_1^k \ldots x_{d_{k+1}-1}^k 1 x_{d_{k+1}+1} \ldots - 0.0^{d_{k+1}-1}10 \ldots \\
&= 0.x_1^k \ldots x_{d_{k+1}-1}^k 0 x_{d_{k+1}+1} \ldots
\end{aligned}$$

This shows that $x_l^{k+1} = x_1^k$ for all $l \neq d_{k+1}$. Now, set

$$\sigma_{k+1} = \mu_{d_{k+1}}^k.$$

We have $|\sigma_{k+1}| = |\mu_{d_{k+1}}^k| = d_{k+1}$. And for any $l \neq d_{k+1}$ with $x_l^{k+1} = 1$, let

$$\mu_l^{k+1} = \mu_l^k.$$

Observe $|\mu_l^{k+1}| = |\mu_l^k| = l$. Note $\mu_{d_{k+1}}^{k+1} \neq 1$, so $x_l^{k+1} = 1 \Leftrightarrow x_l^k = 1$. Hence, $\{\mu_l^{k+1} : x_l^{k+1} = 1\} = \{\mu_l^{k+1} : x_l^k = 1\}$, so we have defined each element of $\{\mu_l^{k+1} : x_l^{k+1} = 1\}$.

Observe that

$$\begin{aligned} S_{k+1} = \{\sigma_i : i \leq k+1\} \cup \{\mu_m^{k+1} : x_m^{k+1} = 1\} &= \{\sigma_i : i \leq k\} \cup \{\sigma_{k+1}\} \cup (\{\mu_m^k : x_m^k = 1\} \setminus \{\mu_{d_{k+1}}^k\} \\ &= \{\sigma_i : i \leq k\} \cup \{\mu_{d_{k+1}}^k\} \cup (\{\mu_m^k : x_m^k = 1\} \setminus \{\mu_{d_{k+1}}^k\}) \\ &= \{\sigma_i : i \leq k\} \cup \{\mu_m^k : x_m^k = 1\}. \end{aligned}$$

The final set is prefix-free by assumption. Moreover, $\mu_{d_{k+1}}^k$ is effectively chosen by assumption, so σ_{k+1} is effectively chosen. Thus, we have continued our construction in this case.

On the other hand, suppose $d_{k+1} = 0$. We claim that some $j \in \mathbb{N}$ with $j < d_{k+1}$ must exist such that $x_j^k = 1$. If no such j exists, then $x_j^k = 0$ for all $j < d_{k+1}$. Consequently,

$$0.x^k = 0.0^{d_{k+1}} x_{d_{k+1}+1}^k \cdots.$$

Note that the exponent on the 0 here means to repeat 0 d_{k+1} times. When an exponent is over a 0 or 1, we mean repeat that symbol. But when we write x^k, this is a naming scheme and does *not* mean to repeat x k times. By construction, $1 - \sum_{i \leq k} 2^{-d_i} = 0.x^k$. Note $x^k \in 2^{<\omega}$, so it does not hold that $x_p^k = 1$ for all $p \geq d_{k+1} + 1$. That would be the equivalent situation to $0.\overline{9} = 1$ in $\mathbb{R}$. Consequently,

$$2^{-d_{k+1}} = 0.0^{d_{k+1}-1}1 > 0.0^{d_{k+1}} x_{d_{k+1}+1}^k \ldots = 1 - \sum_{i \leq k} 2^{-d_i}.$$

Hence, $\sum_{i \leq k+1} 2^{-d_i} > 1$. But $2^{-d_i} \geq 0$ for all $i \in \mathbb{N}$, so $\sum_{i \geq 1} 2^{-d_i} \geq \sum_{i \leq k+1} d^{-d_i} > 1$, which contradicts our assumption. Hence, some $j < d_{d_{k+1}}$ exists such that $x_j^k = 1$. We can effectively pick that largest such j.

Since $j \leq d_{k+1}$ is maximal, $x_q^k = 0$ when $j < q \leq d_{k+1}$. Note that

$$\begin{aligned} 0.x^{k+1} = 1 - \sum_{i \leq k+1} 2^{-d_i} &= 1 - 2^{-d_{k+1}} - \sum_{i \leq k} 2^{-d_i} \\ &= (1 - \sum_{i \leq k} 2^{-d_i}) - 2^{-d_{k+1}} \\ &= 0.x^k - 2^{-d_{k+1}} \\ &= 0.x_1^k \ldots x_{j-1}^k 10^{d_{k+1}-j} x_{d_{k+1}+1-j}^k \ldots - 0.0^{d_{k+1}-1}1 \\ &= 0.x_1^k \ldots x_{j-1}^k 01^{d_{k+1}-j} x_{d_{k+1}+1-j}^k \cdots. \end{aligned}$$

Hence, $x_l^{k+1} = x_l^k$ for all l except when $j \leq l \leq d_{k+1}$. In those cases, $x_j^{k+1} = 0$ and $x_l^{k+1} = 1$. To understand the last equality, for an example consider $0.1000 - 0.0001 = 0.0111$. This is analogous to carrying system of decimals in $\mathbb{R}$ where $0.1000 - 0.0001 = 0.0999$.

Now, $x_m^{k+1} = 1$ if and only if either $x_m^k = 1$ for $m \neq j, \ldots, d_{k+1}$ or $j < m \leq d_{k+1}$. For $m \neq j, \ldots, d_{k+1}$, let

$$\mu_m^{k+1} = \mu_m^k.$$

Note $|\mu_m^{k+1}| = |\mu_m^k| = m$. For $j < m \leq d_{k+1}$, let

$$\mu_m^{k+1} = \mu_j^k 0^{m-j-1}1.$$

Note $|\mu_m^{k+1}| = |\mu_j^k 0^{m-j-1}1| = j + (m - j - 1) + 1 = m$. Finally, define

$$\sigma_{k+1} = \mu_j^k 0^{d_{k+1}-j}.$$

Note $|\sigma_{k+1}| = |\mu_j^k 0^{d_{k+1}-j}| = j + d_{k+1} - j = d_{k+1}$.

We claim that $S_{k+1} = \{\sigma_i : i \leq k+1\} \cup \{\mu_m^{k+1} : x_m^{k+1} = 1\}$ is prefix-free. Observe

$$\begin{aligned} S_{k+1} &= \{\sigma_i : i \leq k+1\} \cup \{\mu_m^{k+1} : x_m^{k+1} = 1\} \\ &= \{\sigma_i : i \leq k\} \cup \{\sigma_{k+1}\} \cup \{\mu_m^{k+1} : x_m^{k+1} = 1\} \\ &= \{\sigma_i : i \leq k\} \cup \{\mu_j^k 0^{d_{n+1}-j}\} \cup \{\mu_m^k : x_m^{k+1} = 1 \text{ and } m \neq j+1, \ldots, d_{k+1}\} \\ &\qquad \cup \{\mu_j^k 0^{m-j-1} 1 : j < m \leq d_{k+1}\}. \end{aligned}$$

We will show that S_{k+1} is prefix-free in parts. First, set $A = \{\mu_j^k 0^{m-j-1} 1 : j < m < d_{k+1}\}$. Pick any $\tau = \mu_j^k 0^{u-j-1} 1, \tau' = \mu_j^k 0^{v-j-1} 1 \in A$. Suppose $\mu_j^k 0^{u-j-1} 1 \preceq \mu_j^k 0^{v-j-1} 1$. Then $0^{u-j-1} 1 \preceq 0^{v-j-1} 1$. This holds only if $u = v$, which implies $\tau = \tau'$. Hence, A is prefix-free.

Next, set $B = A \cup \{\sigma_{k+1}\}$. Let $\mu_j^k 0^{u-j-1} 1 \in A$ be arbitrary. Then $j < u < d_{k+1}$. First, suppose $\mu_j^k 0^{u-j-1} 1 \preceq \sigma_{k+1} = \mu_j^k 0^{d_{k+1}-j}$. Then $0^{u-j-1} 1 \preceq 0^{d_{k+1}-j}$. Clearly, this is not possible. Otherwise, $\mu_j^k 0^{d_{k+1}-j} \preceq \mu_j^k 0^{u-j-1} 1$, which implies $0^{d_{k+1}-j} \preceq 0^{u-j-1} 1$. But then $d_{k+1} - j \leq u - j - 1$, so $d_{k+1} < d_{k+1} + 1 < u$, which is a contradiction. Hence, B is prefix-free.

Continuing, let $C = B \cup \{\mu_m^k : x_m^{k+1} = 1 \text{ and } m \neq j+1, \ldots, d_{k+1}\}$. Note all elements B have the form $\mu_j^k \tau$ for some $\tau \in 2^{<\omega}$. Let $\mu_l^k \in \{\mu_l^k : x_l^{k+1} = 1 \text{ and } l \neq j+1, \ldots, d_{k+1}\}$ be such that $\mu_l^k \preceq \mu_j^k$ or $\mu_j^k \preceq \mu_l^k$. Since S_k is prefix-free, $j = l$. But $j \neq l$ by construction since $l < j$ or $l > d_{k+1}$. Hence, C is prefix-free.

Lastly, let's show that $S_{k+1} = C \cup \{\sigma_i : i \leq k\}$ is prefix-free. Note all elements in C have the form $\mu_m^k \tau$ for some $m \in \mathbb{N}$ and (possible empty string) $\tau \in 2^{<\omega}$. But if $\mu_m^k \tau \preceq \sigma_i$ or $\sigma_i \preceq \mu_m^k \tau$, then $\mu_m^k \preceq \sigma_i$ or $\sigma_i \preceq \mu_m^k$. But $\sigma_i \mid \mu_m^k$ for all m since S_k is prefix-free. Hence, S_{k+1} is prefix-free.

Therefore, we have effectively defined σ_i for each $i \in \omega$ with $|\sigma_i| = d_i$. Since each σ_i is chosen effectively from (d_i, τ_i), which is computable by assumption, there exists a Turing machine M defined by $M(\sigma_i) = M(\tau_i)$ for all $i \in \omega$. Note $\mathrm{dom}(M) \subseteq \bigcup_{k \geq 1} S_k$, so M is the desired prefix-free Turing machine. $\dashv$

Remark 2.10. We say that the *weight* of a request (d, τ) is 2^{-d}. Moreover, the weight of a computable sequence of requests $(d_i, \tau_i)_{i \in \omega}$ is $\sum_{i \geq 1} 2^{-d_i}$. The set of sequences of requests is called a *KC-set* if $\sum_{i \geq 1} 2^{-d_i} \leq 1$.

The following corollary shows how the KC Theorem allows us to put a rough bound on K.

Corollary 2.11.
1. If a sequence of requests $(d_i, \tau_i)_{i \in \omega}$ has a finite weight, then there exists some constant $c \in \mathbb{R}$ such that $(d_i + c, \tau_i)$ is a KC-set.
2. There exists a $d_i \in \mathbb{N}$ for each $\tau_i \in 2^{<\omega}$ such that (d_i, τ_i) is a KC-set.
3. If (d_i, τ_i) is a KC-set, then $K(\tau_i) \leq d_i + c$ for some constant $c \in \mathbb{N}$ for all $i \in \omega$.

Proof. 1. Suppose $(d_i, \tau_i)_{i \in \omega}$ has a finite weight less than or equal to some effectively chosen $b \in \mathbb{R}$. Then $\sum_{i \geq 0} 2^{-d_i} \leq b < \infty$. Hence, $\sum_{i \geq 0} \frac{2^{-d_i}}{b} \leq 1$. Without loss of generality, assume $b > 1$ since otherwise

$$\sum_{i \geq 0} 2^{-d_i} \leq \sum_{i \geq 0} \frac{2^{-d_i}}{b} \leq 1.$$

Let c be the least integer greater than or equal to b. Then $2^c \geq b$. Then

$$\sum_{i \geq 0} 2^{-(d_i + c)} = \sum_{i \geq 0} 2^{-d_i} 2^{-c} = \sum_{i \geq 0} \frac{2^{-d_i}}{2^c} \leq \sum_{i \geq 0} \frac{2^{-d_i}}{b} \leq 1.$$

Note c was chosen effectively from b, so $(d_i + c, \tau_i)$ is a KC-set.

2. Let $\{\tau_i\}_{i\in\omega}$ be an effective enumeration of $2^{<\omega}$ where $\tau_0 = \lambda$, the empty string. Set $d_0 = 2$ and $d_i = |\tau_i| + 2\log_2|\tau_i| + 3$ for $i \geq 1$. Our definition of d_i for $i \geq 1$ is well-defined since $|\tau_i| \neq 0$. Observe that for each $n \in \mathbb{N}$ there are 2^n strings σ such that $|\sigma| = n$. Consequently,

$$\begin{aligned}
\sum_{i\geq 0} 2^{-d_i} &= 2^{-d_0} + \sum_{i\geq 1} 2^{-d_i} \\
&= \frac{1}{4} + \sum_{i\geq 1} 2^{-(|\tau_i| + 2\log_2|\tau_i| + 3)} \\
&= \frac{1}{4} + \sum_{i\geq 1} 2^{-|\tau_i|} 2^{\log_2(|\tau_i|^{-2})} 2^{-3} \\
&= \frac{1}{4} + \sum_{i\geq 1} \frac{2^{-|\tau_i|}}{8|\tau_i|^2} \\
&= \frac{1}{4} + \sum_{n\geq 1} 2^n \frac{2^{-n}}{8|n|^2} \text{ by our observation} \\
&= \frac{1}{4} + \frac{1}{8}\sum_{n\geq 1} \frac{1}{n^2} \\
&= \frac{1}{4} + \frac{1}{8}\frac{\pi^2}{6} \text{ by series convergence where } \pi \text{ denotes the number not a string} \\
&\leq 1.
\end{aligned}$$

Therefore, (d_i, τ_i) is a KC-set

3. Let (d_i, τ_i) be a KC-set. Then there exists a prefix-free machine M such that $M(\sigma_i) = \tau_i$ with $|\sigma_i| = d_i$ for all $i \in \omega$. So by 2.7, $K(\tau_i) \leq K_M(\tau_i) + c = \min\{\sigma : M(\sigma) = \tau_i\} + c \leq d_i + c$ for some constant $c \in \mathbb{N}$. $\dashv$

At the outset of this section, we expressed an interest in the "information" of strings. We will now introduce a way to formalize this notion.

Definition 2.12. A partial function $F : 2^{<\omega} \to \mathbb{N}$ is said to be an *information content measure* if both

1. $\sum_{\sigma\in\mathrm{dom}(F)} 2^{-F(\sigma)} \leq 1$, and
2. $\{(\sigma, k) : k \in \mathbb{N} \text{ and } F(\sigma) \leq k\}$ is a computably enumerable set.

The following results will further characterize prefix-free Kolmogorov complexity. They establish that K is a minimal information content measure.

Proposition 2.13. We have $\sum_{\sigma\in 2^{<\omega}} 2^{-K(\sigma)} \leq 1$.

Proof. By part two corollary 2.11, there is a KC-set $(d_i, \sigma_i)_{i\in\omega}$ where $\{\sigma_i\}_{i\in\mathbb{N}}$ is an enumeration of $2^{<\omega}$. So by part three of corollary 2.11, there exists a constant $c \in \mathbb{N}$ such that $K(\sigma_i) \leq d_i + c$ for

each $i \in \omega$. If $c \leq 0$, then $K(\sigma_i) \leq d_i + c \leq d_i$ for each i. Consequently, $2^{-K(\sigma_i)} \leq 2^{-d_i}$. Otherwise, $c > 0$, which implies $2^{-c} < 1$. In this case for each $i \in \omega$ we have

$$K(\sigma_i) \leq d_i + c \Rightarrow 2^{-K(\sigma_i)} \leq 2^{-d_i - c} = 2^{-d_i} 2^{-c} \leq 2^{-d_i}.$$

Hence, $\sum_{\sigma \in 2^{<\omega}} 2^{-K(\sigma_i)} \leq \sum_{i \in \omega} 2^{-d_i}$. Since $(d_i, \sigma_i)_{i \in \mathbb{N}}$ is a KC-set, $\sum_{\sigma \in 2^{<\omega}} 2^{-K(\sigma_i)} \leq 1$. $\dashv$

The above result implies the first condition for K to be an information content measure since $\operatorname{dom} K \subseteq 2^{<\omega}$. We only have to show that K meets the second condition.

Proposition 2.14. K is an information-content measure.

Proof. It is not difficult to show that K meets the two conditions of an information content measure.

1. This follows from proposition 2.13 as just mentioned.

2. Let's show that $B = \{(\sigma, k) : k \in \mathbb{N} \text{ and } K(\sigma) \leq k\}$ is computably enumerable. Given $k \in \mathbb{N}$, note that there are only finitely many strings τ with $|\tau| \leq k$. And $(\sigma, k) \in B \Leftrightarrow \mathcal{U}(\tau) = \sigma$ for some $|\tau| \leq k$. Hence, the following partial function f is computable:

$$f(\sigma, k) = \begin{cases} 1 & \text{if } K(\sigma) \leq k, \\ \uparrow & \text{otherwise.} \end{cases}$$

   Note $B = \operatorname{dom} f$. Thus, by proposition 1.11, B is computably enumerable. $\dashv$

The reason we are interested in information content measures is that K has a particularly nice property as one. Namely, K is a minimal i.c.m. Given the unruliness of prefix-free complexity, a common strategy is to find ways to bound it. With the following theorem, we only need to show that a function is an information content measure put a bound on K up to a constant.

Theorem 2.15. Let F be an information content measure. Then there exists a constant $c \in \mathbb{N}$ such that for all $\sigma \in \operatorname{dom}(F)$ we have $K(\sigma) \leq F(\sigma) + c$. In other words, K is a minimal information content measure.

Proof. First, we will show that each information content measure induces a prefix-free machine. Let F be any information content measure. Then $\{(\sigma, k) : F(\sigma) \leq k\}$ is computably enumerable by definition. Hence, we can effectively enumerate this set $(\sigma_i, k_i)_{i \in \omega}$. Then $R = \{(k_i + 1, \sigma_i) : F(\sigma_i) \leq k_i\}$ is c.e. since the functions defined $(a, b) \mapsto (b, a)$ and $a \mapsto a + 1$ are intuitively computable. So R is a computable sequence of pairs; i.e., given $i \in \omega$, we can compute $(k_i + 1, \sigma_i) \in R$. Consequently,

$$\sum_{(k+1,\sigma) \in R} 2^{-(k+1)} < \sum_{(k+1,\sigma) \in R} 2^{-k} \leq \sum_{F(\sigma)\downarrow} 2^{-F(\sigma)} \leq 1.$$

Thus, R is a KC-set. So by the KC Theorem, there exists a prefix-free Turing machine M where for each $\sigma \in \operatorname{dom} F$ there exists τ such that $|\tau| = F(\sigma) + 1$ and $M(\tau) = \sigma$.

To show that K is minimal, take any information content measure F and any $\sigma \in 2^{<\omega}$. For M as constructed above, $K_M(\sigma) = \min\{|\tau| : M(\tau) = \sigma\} \leq F(\sigma) + 1$. And by proposition 2.7, $K(\sigma) \leq K_M(\sigma) + c$ for some $c \in \mathbb{N}$. Therefore, $K(\sigma) \leq F(\sigma) + c + 1$. $\dashv$

We can now give our first attempt at randomness.

Definition 2.16. A sequence $A \in 2^\omega$ is said to be 1-*Random* if there exists some constant $c \in \mathbb{N}$ such that for all $n \in \mathbb{N}$ we have $K(A \upharpoonright n) \geq n - c$.

Remark 2.17. By the above definition, only infinite sequences can be random. To understand this, note that a string $\sigma \in 2^{<\omega}$ can be viewed as the sequence $\sigma 0000\ldots$ where we only need the "information" of σ to know the entire sequence. We thus consider randomness to be a property only of infinite sequences.

This definition says that the more we want Turing machines to tell us about a random string, the more information we will have to give the machine. This corresponds to our intuition that a random string cannot be be described all at once unless we already know its description. Nevertheless, this definition seems mysterious.[3]

3 Approaching Randomness with Measure Theory

Before computability theory, mathematicians would look for distinctive characteristics of random sequences. Some noticed that a random sequence must have the same proportion of ones and zeros since if a fair coin was continually flipped, we would expect the number of heads and tails to be roughly equal. However, this condition is not sufficient as the non-random sequence $0101010101\ldots$ satisfies it. We could add more conditions to rule out such sequences, but it is almost impossible to rule out all counterexamples when randomness is not already formally defined. Given any list of conditions, there is a good chance that new ones will be added in the future, yet we do not want a definition of randomness that is likely to change over time. Instead of listing particular conditions, Martin-Löf developed a general notion for a test that random sequences always must fail.

These statistical tests will use measure theory in addition to computability. Following Simpson, we will only briefly deal with results from measure theory and instead focus on the computability side [**?**]. Measure theory requires viewing 2^ω with a topology called the *Cantor space*.

Proposition 3.1. The collection of sets $[\sigma] = \{X \in 2^\omega : \sigma \prec X\}$ for all $\sigma \in 2^{<\omega}$ is the basis of a topology on 2^ω.

Proof. Let $\mathcal{B}$ denote this collection. For any $X \in 2^\omega$, we have $(X \upharpoonright 1) \in 2^{<\omega}$ and $X \in [X \upharpoonright 1]$. Second, take any $[\sigma], [\tau] \in \mathcal{B}$. Suppose we have $X \in [\sigma] \cap [\tau]$. Then we have $X \in [\sigma]$ and $X \in [\tau]$. Also, we know either $\sigma \preceq \tau$ or $\tau \preceq \sigma$ since otherwise $[\sigma] \cap [\tau] = \varnothing$. Without loss of generality, suppose $\sigma \preceq \tau$. Then $[\tau] \subseteq [\sigma]$, so $X \in [\tau] \subseteq [\sigma] \cap [\tau]$. Thus, $\mathcal{B}$ is in fact a basis. $\dashv$

On a topological space, we define a measure as follows.

Definition 3.2. For a nonempty set I, let $\mathcal{P}(I)$ be the power set of I. A set $\mathcal{S} \subseteq \mathcal{P}(I)$ is said to be a *σ-algebra* on I if we have

1. $\varnothing \in \mathcal{S}$ and $I \in \mathcal{S}$ and
2. $\mathcal{S}$ is closed under countable unions, countable intersections, and complementations.

We say the function $\mu : \mathcal{S} \to [0, 1]$ is a *probability measure* for a σ-algebra $\mathcal{S}$ if

1. $\mu(\varnothing) = 0$,
2. $\mu(I) = 1$,
3. (Countable additivity) $\mu(\bigcup_{n=1}^{\infty} S_n) = \sum_{n=1}^{\infty} \mu(S_n)$ for any pairwise disjoint sets $S_1, S_2, \ldots \in \mathcal{S}$, and
4. (Monotonicity[4]) If $S_1 \subseteq S_2 \subseteq I$, then $\mu(S_1) \leq \mu(S_2)$.

[3] We might ask why we should use prefix-free complexity instead of plain complexity. In fact, it can be shown that no sequence satisfies our criterion if we instead use plain complexity [**?**].

[4] In fact, monotonicity follows from the other three properties.

We call the ordered triple $(I, \mathcal{S}, \mu)$ as given above a *probability space*, and we say the *measure* of $X \in \mathcal{S}$ is $\mu(X)$.

Definition 3.3. The smallest σ-algebra of a space I containing the open sets is said to be the collection $\mathcal{C}$ of *Borel sets* of I. A measure $\mu : \mathcal{C} \to [0, 1]$ is said to be a *Borel probability measure on I.*

We are interested in a specific measure on 2^ω. Intuitively, we view sequences $X \in 2^\omega$ as the result of independent coin flips. In other words, we flip a fair coin infinitely many times and set $X(n) = 0$ if the nth flip is tails and set $X(n) = 1$ otherwise. As such coin flips give our paradigm examples of random sequences, this measure is clearly relevant.

Theorem 3.4. There exists a Borel probability measure on 2^ω such that $\mu([\sigma]) = 2^{-|\sigma|}$ for all $\sigma \in 2^{<\omega}$.

Proof. The reader can consult section 1.9 of Nies's *Computability and Randomness* [**?**] for a proof. In the proof, we first define an *outer measure* μ^* on the power set of 2^ω. Then we show that that Borel sets satisfy a property called *measurability.* Namely, $A \subseteq 2^\omega$ is measurable if for all $X \subseteq 2^\omega$ we have

$$\mu^*(X) = \mu^*(X \cap A) + \mu^*(X \cap A^c).$$

We define μ to be the restriction of μ^* to the Borel sets. Using the measurability of these sets, we can prove that μ satisfies the properties of a measure. $\dashv$

While this framework seems rather unlike Kolmogorov, we have already considered the numbers such as $2^{-|\sigma|}$ in the previous section. Moreover, measure theory alone cannot characterize randomness without some computability theory.

Definition 3.5. We say a set $U \subseteq 2^\omega$ is Σ^0_1 if $U = \{X \in 2^\omega : \exists k[R(X, k)]\}$ for some computable relation R. Furthermore, we say a sequence of sets $\{U_k\}_{k\in\omega}$ is *uniformly* Σ^0_1 if there exists some computable relation R such that for all $k \in \mathbb{N}$ we have

$$U_k = \{X \in 2^\omega : \exists n R(X, n, k)\}.$$

Another similarity to the definition of 1-randomness above is the relation of Σ^0_1 classes to Turing machines.

Proposition 3.6. A set $U \in 2^\omega$ is Σ^0_1 if and only if it is computably enumerable.

Proof. [**?**] ($\Rightarrow$) Suppose $U \subseteq 2^\omega$ is Σ^0_1. Then $U = \{X \in 2^\omega : \exists k[R(X, k)]\}$ for some computable relation R. Define $f : \mathbb{N} \to \mathbb{N}$ by $f(x) = 0$ if $\exists k[R(x, k]$ and $f(x) \uparrow$ otherwise. Note

$$x \in \operatorname{dom} f \Leftrightarrow \exists k[R(x, k)] \Leftrightarrow x \in U.$$

Hence, $\operatorname{dom} f = U$.

We can compute f with the following algorithm. For some input x, we can effectively check if $R(x, k)$ holds for each k since R is computable. If $R(x, k)$ halts for some k, then $f(x) = 0$. Otherwise, $R(x, k)$ halts for no k, so $f(x) \uparrow$. So by the Turing thesis, f is partial computable. Hence, U is the domain of a p.c. function. Thus, by proposition 1.11, U is computably enumerable.

($\Leftarrow$) Next, suppose U is computably enumerable. Then either $U = \varnothing$ or $U = \operatorname{im} f$ for some p.c. function f. First, suppose $U = \varnothing$. Note that the relation $R(x, k)$ given by the set $\{(x, k) : x = x + 1\}$ is computable by any algorithm that never halts such as a infinite loop program. And $\varnothing = \exists k[R(x, k)]$. So $\varnothing$ is Σ^0_1.

Otherwise, we have $U = \operatorname{im} f$ for some p.c. function f. Define the relation R by

$$(x, s) \in R \Leftrightarrow x \in f[s].$$

To see that R is computable, define a Turing machine $M'(x, s)$ to simulate M on input x for s stages. If M halts, then output 1. Otherwise, output 0. Then $M'(x, s)$ halts if and only if $(x, s) \in R$. Hence, R is computable. Note $x \in U = \operatorname{im} f \Leftrightarrow \exists s[R(x, s)]$. Thus, R is Σ^0_1. $\dashv$

Remark 3.7. Analogously, a collection of sets $\{A_n\}_{n\in\omega}$ is uniformly c.e. if and only if it is uniformly Σ^0_1.

We can now combine this definition from computability with measure theory to a define Martin-Löf test.

Definition 3.8. A set $T \subseteq 2^\omega$ is said to be *null* if $\mu(T) = 0$. Moreover, we say T is *effectively null* if there exists a uniformly Σ^0_1 sequence $\{U_k\}_{k\in\omega}$ in 2^ω where $\mu(U_k) \leq 2^{-k}$ for all $k \in \mathbb{N}$ and $T \subseteq \bigcap_{k\in\omega} U_k$. We call the sequence $\{U_k\}$ a *Martin-Löf Test*.

The two parts of a Martin-Löf test generalize the conditions random sequences should not satisfy. We require the sets U_k to be uniformly Σ^0_1 because random sequences should not be computable. And we require $\mu(U_k) \leq 2^{-k}$ so that as k increases the set U_k is making a bolder claim about the sequence. These tests are all we need to characterize randomness.

Definition 3.9. A sequence $X \in 2^\omega$ is *Martin-Löf Random* if for all Martin-Löf Tests $\{U_k\}_{k\in\omega}$ we have that $X \notin \bigcap_{k\in\omega} U_k$. In other words, X is Martin-Löf random if X is not effectively null.

4 Approaching Randomness with Martingales

We now turn to yet another framework, which *prima faceii* is unlike complexity or measure theory. Our inspiration comes from gambling. Random sequences are those that cannot be predicted without luck. First, we will formalize the notion of a betting strategy.

Definition 4.1. A function $B : 2^{<\omega} \to \overline{R}_+$ is said to be a *martingale* if for every $\sigma \in 2^{<\omega}$ we have

$$B(\sigma) = \frac{B(\sigma 0) + B(\sigma 1)}{2}.$$

For the empty string λ, we call $B(\lambda)$ the *initial capital* of B.

Suppose the croupier flips a coin. The gambler, attracted by the favorable odds, plays aggressively by placing bets equal to his entire capital on each flip. For example, if he starts out with \$100, then he might bet \$60 on the next flip being heads and \$40 on it being tails. After a flip, he wins the amount bet on that outcome and loses the amount bet against that outcome. The martingale function keeps track of how much money the gambler would have at any possible point in the game. Let's consider some examples.

Example 4.2. [**?**]

1. The constant function B defined by $B(\sigma) = 1$ for all $\sigma \in 2^{<\omega}$ is a martingale. To see this, note $B(\sigma) = 1 = (1 + 1)/2 = (B(\sigma 0) + B(\sigma 1))/2$. This corresponds to the betting strategy where the gambler starts with \$1.00 puts half his money on heads and the other half on tails each flip.

2. For more slightly interesting Martingale, take a fixed $\sigma \in 2^{<\omega}$ and set

$$B_\sigma(\tau) = \begin{cases} 2^{|\tau|} \text{ if } \tau \prec \sigma, \\ 2^{|\sigma|} \text{ if } \sigma \preceq \tau, \\ 0 \text{ if } \sigma \mid \tau. \end{cases}$$

Note $\tau \prec \tau 0$ and $\tau \prec \tau 1$. So if $\sigma \preceq \tau$, then $\sigma \preceq \tau 0$ and $\sigma \preceq \tau 1$. Hence,

$$B_\sigma(\tau) = 2^{|\sigma|} = \frac{2^{|\sigma|+1}}{2} = \frac{2^{|\sigma|} + 2^{|\sigma|}}{2} = \frac{B_\sigma(\tau 0) + B_\sigma(\tau 1)}{2}.$$

And if $\tau \mid \sigma$, then $\tau 0 \mid \sigma$ and $\tau 1 \mid \sigma$. Consequently,

$$B_\sigma(\tau) = 0 = \frac{0+0}{2} = \frac{B_\sigma(\tau 0) + B_\sigma(\tau 1)}{2}.$$

Otherwise, we have $\tau \prec \sigma$. Without loss of generality, suppose $\tau 0 \preceq \sigma$. Then $\tau 1 \mid \sigma$. Hence,

$$B_\sigma(\tau) = 2^{|\tau|} = \frac{2^{|\tau|+1}}{2} = \frac{2^{|\tau 0|} + 0}{2} = \frac{B_\sigma(\tau 0) + B_\sigma(\tau 1)}{2}.$$

Thus, B_σ is in fact a martingale. This betting strategy could be thought of as using a "lucky string". We bet our entire capital based on what σ tells us, and we bet nothing on strings that are incomparable with σ.

In practice, gamblers might not bet their entire capital on each flip. A typical scenario is a gambler who tips those around her for good luck. This would not be a relevant observation if its corresponding notion was not useful. However, it is.

Definition 4.3. A function $S : 2^{<\omega} \to \overline{R}_+$ is said to be a *supermartingale* if for every $\sigma \in 2^{<\omega}$ we have

$$S(\sigma) \geq \frac{S(\sigma 0) + S(\sigma 1)}{2}.$$

As above, we call $S(\lambda)$ the initial capital of S.
We can define the *charity function* $d : 2^{<\omega} \to \overline{R}_+$ of a supermartingale S by

$$d(\sigma) = S(\sigma) - \frac{S(\sigma 0) + S(\sigma 1)}{2}.$$

Remark 4.4. Note that each martingale is also a supermartingale where equality always holds.

Supermartingales are useful because it is often easier to prove an inequality than an equality. In fact, we will show for our considerations that martingales and supermartingales are equivalent.

We are interested in when (super)martingales succeed on an infinite sequence. Intuitively, success means predicting the coin flips well enough that we can break the bank of the casino by playing long enough. In our theoretical setting, this might take longer than the universe has existed of course. We need the notion of the limit supremum to define success.

Definition 4.5. For a sequence $\{x_n\}_{n\in\omega}$ in $\mathbb{R}$, the *limit supremum* of the sequence is its largest limit point. We denote the limit supremum of a sequence by $\limsup_n x_n$. If the sequence has a subsequence diverging to ∞, we say $\limsup x_n = \infty$.

In general, $\limsup x_n \neq \sup x_n$ as the following examples will illustrate.

Example 4.6. [?]

1. The sequence $\{1/n\}_{n\in\omega}$ converges to 0, so it has only one limit point 0. Hence, $\limsup_n 1/n = 0$. However, $\sup_n 1/n = 1$.
2. The sequence $\{-1^n\}$ has -1 and 1 has its only limit points. Hence, $\limsup_n (-1)^n = \sup\{-1, 1\} = 1$. In this case, we also have $\sup_n(-1^n) = 1$.
3. Define the sequence $\{x_n\}_{n\in\omega}$ by $x_n = n$ for n even and $x_n = 0$ for n odd. Then $\limsup_n x_n = \infty$.

Now we can define feature of (super)martingales that will be used later to characterize randomness.

Definition 4.7. For a martingale B and a sequence $X \in 2^\omega$, we define $B(X)$ by

$$B(X) = \limsup_n B(X \upharpoonright n).$$

We define the *success set* of B to be $\mathrm{Succ}(B) = \{X \in 2^\omega : B(X) = \infty\}$. We say that B *succeeds* on X if $X \in \mathrm{Succ}(B)$.

For a supermartingale S and a string $X \in 2^\omega$, we define $S(X)$ and $\mathrm{Succ}(S)$ analogously.

Observe that the gambler need not be perfect at prediction in order to have a successful strategy for a sequence. We will show that martingales and supermartgales are the same insofar as success sets are concerned.

Proposition 4.8. For every supermartingale S, there exists a martingale B such that for all $\sigma \in 2^{<\omega}$ we have $B(\sigma) \geq S(\sigma)$ and $S(\lambda) = B(\lambda)$.

Proof. [?] Take an arbitrary supermartingale S with charity function d. Define $B : 2^{<\omega} \to \overline{R}_+$ by

$$B(\sigma) = S(\sigma) + \sum_{\tau \prec \sigma} d(\tau).$$

Since $\sum_{\tau\prec\sigma} d(\tau) \geq 0$, we have $B(\sigma) \geq S(\sigma)$ for all $\sigma \in 2^{<\omega}$. And $\sum_{\tau\prec\lambda} d(\tau) = 0$, so $B(\lambda) = S(\lambda) + 0 = S(\lambda)$.

Next, let $\pi \in 2^{<\omega}$ be arbitrary. Then

$$\begin{aligned}
B(\pi 0) + B(\pi 1) &= S(\pi 0) + \sum_{\tau\prec\pi 0} d(\tau) + S(\pi 1) + \sum_{\tau\prec\pi 1} d(\tau) \\
&= S(\pi 0) + S(\pi 1) + 2\sum_{\tau\preceq\pi} d(\tau) \\
&= S(\pi 0) + S(\pi 1) + 2d(\pi) + 2\sum_{\tau\prec\pi} d(\tau) \\
&= 2S(\pi) + 2\sum_{\tau\prec\pi} d(\tau) \\
&= 2B(\pi).
\end{aligned}$$

Therefore, B is a martingale that satisfies the desired properties. ⊣

Corollary 4.9. For every supermartingale S, there exists a martingale M such that $\mathrm{Succ}(S) \subseteq \mathrm{Succ}(M)$.

Proof. Let S be any supermartingale. Then there exists a martingale B such that $B(\lambda) = S(\lambda)$ and $S(\sigma) \leq B(\sigma)$ for all $\sigma \in 2^{<\omega}$ by proposition 4.8. So for any $X \in 2^\omega$ and $n \in \omega$, we have $S(X \upharpoonright n) \leq B(X \upharpoonright n)$. Consequently,

$$\limsup_n (S(X \upharpoonright n)) \leq \limsup_n (B(X \upharpoonright n)).$$

Hence, $S(X) = \infty \Rightarrow B(X) = \infty$. Thus, $\mathrm{Succ}(S) \subseteq \mathrm{Succ}(B)$. $\dashv$

Because of this corollary and remark 4.4, we can interchangeably use martingales and supermartingales when we are only interested in their success sets. Consequently, we will only deal with martingales for the rest of this paper, but the results also will apply to supermartingales.

The next result allows us to construct martingales.

Proposition 4.10. 1. If B is a martingale, then rB is a martingale for all $r \in \overline{\mathbb{R}}_+$.

2. If $\{B_n\}_{n\in\omega}$ is a countable collection of martingales such that $\sum_{n\in\omega} B_n(\lambda) < \infty$, then $\sum_{n\in\omega} B_n$ is also a martingale.

Proof. [**?**]

1. Let B be a martingale. Then for all $\sigma \in 2^{<\omega}$ we have $B(\sigma) = \frac{B(\sigma 0)+B(\sigma 1)}{2} \geq 0$. Hence, $rB(\sigma) = r\frac{B(\sigma 0)+B(\sigma 1)}{2} = \frac{rB(\sigma 0)+rB(\sigma 1)}{2}$ for all $r \in \mathbb{R}$. And if $r \in \overline{\mathbb{R}}_+$, then $rB(\sigma) \geq 0$. Thus, rb is a martingale for $r \in \overline{\mathbb{R}}_+$.

2. Next, let $\{B_n\}_{n\in\omega}$ be a collection of martingales such that $\sum_{n\in\omega} B_n(\lambda) < \infty$. Let

$$B = \sum_{n\in\omega} B_n.$$

By induction on $|\sigma|$, we can show that $B(\sigma) < \infty$ for all $\sigma \in 2^{<\omega}$. The base case $B(\lambda) < \infty$ follows by assumption. Suppose all strings τ of length $|\sigma|$ are such that $B(\tau) < \infty$. Then

$$\begin{aligned} B(\tau 0) + B(\tau 1) &= \sum_{n\in\omega} B_n(\tau 0) + \sum_{n\in\omega} B_n(\tau 1) \\ &= \sum_{n\in\omega} (B_n(\tau 0) + B_n(\tau 1)) \\ &= \sum_{n\in\omega} 2B_n(\tau) \\ &= 2B(\tau) < \infty. \end{aligned}$$

Thus, $B(\tau 0), B(\tau 1) \leq B(\tau 0) + B(\tau 1) < \infty$. Hence, B is well-defined. In fact, the above equality also shows that B is a martingale. $\dashv$

With the next theorem, we find a connection between martingales, prefix-free sets, and the measure defined in the previous section.

Theorem 4.11. (Kolmogorov's Inequality) Let B be a martingale.

1. If $\sigma \in 2^{<\omega}$ and S is a prefix-free set of extensions of σ, then

$$\sum_{\tau\in S} 2^{-|\tau|} B(\tau) \leq 2^{-|\sigma|} B(\sigma).$$

2. If $R_k = \{\sigma : B(\sigma) \geq k\}$, then $\mu([R_k]) \leq \frac{B(\lambda)}{k}$.

Proof. [?]

1. Let B be an arbitrary martingale.. First, let's show that the desired equality holds when the set S of extensions of a string σ is finite. We can induct on the size of S. For the base case, we have $S = \{\tau\}$ where $\sigma \preceq \tau$ for some $\sigma, \tau \in S$. Let $k = |\tau| - |\sigma|$. Since B is a martingale, we have
$$2^k B(\sigma) = B(\tau) + B(\tau_1) + B(\tau_2) + \ldots + B(\tau_{2k})$$
where the τ_i are all possible extensions of σ other than τ with $|\tau_i| = |\tau|$. Hence, $B(\tau) \leq 2^{|\tau|-|\sigma|}B(\sigma)$, so $2^{-|\tau|}B(\tau) \leq 2^{-|\sigma|}B(\sigma)$.

   For the induction hypothesis, suppose the desired inequality holds for all sets with n elements that are extensions of some $\pi \in 2^{<\omega}$. Take an arbitrary set $S \subseteq 2^{<\omega}$ with $n+1$ elements that are extensions of some string σ. Let $\nu \in 2^{<\omega}$ be the longest string such that for all $\tau \in S$ we have $\nu \preceq \sigma$. Such a string ν must exist since all elements of S are extensions of σ. For $i = 0$ and $i = 1$, let $S_i = \{\tau \in S : \nu i \preceq \tau\}$. We claim $|S_i| < n+1$. Otherwise, $|S_i| = n+1$, which implies $S = S_i$. But all elements in S_i extend νi. Hence, $\nu = \nu i$ by choice of ν, which is a contradiction. Hence, $|S_i| \leq n$. It follows by the induction hypothesis that $\sum_{\tau \in S_i} 2^{-|\tau|}B(\tau) \leq 2^{-|\nu i|}B(\nu i) = 2^{-(|\nu|+1)}B(\nu i)$. Since S is a prefix-free set of at least one extensions of ν, we know $\nu \notin S$. Consequently, $S = S_0 \cup S_1$. Hence,
$$\begin{aligned}
\sum_{\tau \in S} 2^{-|\tau|}B(\tau) &= \sum_{\tau \in S_0} 2^{-|\tau|}B(\tau) + \sum_{\tau \in S_1} 2^{-|\tau|}B(\tau) \\
&\leq 2^{-(|\nu|+1)}B(\nu 0) + 2^{-(|\nu|+1)}B(\nu 1) \\
&= 2^{-(|\nu|+1)}(B(\nu 0) + B(\nu 1)) \\
&= 2^{-|\nu|}\frac{B(\nu 0) + B(\nu 1)}{2} \\
&= 2^{-|\nu|}B(\nu) \text{ by the definition of a martingale} \\
&\leq 2^{-|\sigma|}B(\sigma) \text{ by the base case above.}
\end{aligned}$$

   Thus, by induction, $\sum_{\tau \in S} 2^{-|\tau|}B(\tau) \leq 2^{-|\sigma|}B(\sigma)$ for all finite and prefix-free sets S of extensions of a string σ.

   For a contradiction, suppose that for some infinite and prefix-free S of extensions of σ we have
$$\sum_{\tau \in S} 2^{-|\tau|}B(\tau) > 2^{-|\sigma|}B(\tau).$$
   But $2^{-|\tau|}B(\tau) \geq 0$ for all $\tau \in S$. So there is some finite subset $S' \subset S$ such that
$$\sum_{\tau \in S'} 2^{-|\tau|}B(\tau) > 2^{-|\sigma|}B(\sigma).$$
   As S' is a finite and prefix-free set of extensions of σ, this is a contradiction. Therefore, the desired inequality holds for all prefix-free sets S of extensions of a string σ.

2. Next, let $R_k = \{\sigma : B(\sigma) \geq k\}$. Since $R_k \subseteq 2^{<\omega}$, there exists a prefix-free set P such that $[P] = [R_k]$. Note that all strings in P extend λ. Hence,

$$\begin{aligned}\mu(R_k) = \mu([P]) &= \sum_{\tau \in P} 2^{-|\tau|} \text{ since each set } [\tau] \text{ is disjoint for } P \text{ is prefix-free} \\ &\leq \sum_{\tau \in P} \frac{B(\tau)}{k} 2^{-|\tau|} \text{ since each } \frac{B(\tau)}{k} \geq 1 \text{ by the definition of } R_k \\ &\leq \frac{B(\lambda)}{k} \text{ by part (1).}\end{aligned}$$

Thus, $\mu([R_k]) \leq \frac{B(\lambda)}{k}$. $\dashv$

We are now ready for our last stab at randomness. Intuitively, we want no reasonable betting strategy to succeed on a random sequence. Of course, some betting strategies might uncannily succeed on a random sequence due to luck, so we need to exclude such martingales. We will make rigorous this idea of an "unlucky martingale" by requiring them to be computably enumerable. So far, we have only considered computability in terms of natural numbers and finite strings, but martingales are real-valued functions. There is fairly straightforward way to extend our concepts using the concept from analysis that real numbers are sequences of rationals.

Remark 4.12. The reader might even worry that rationals are not covered by our definition. However, note $x \in \mathbb{Q} \Leftrightarrow x = \frac{a}{b}$ for some $a, b \in \mathbb{N}$ with $b \neq 0$. For the purposes of computability, we can treat $\frac{a}{b}$ as $(a, b) \in \mathbb{N}^2$. So a function $f : \mathbb{N}^n \to \mathbb{Q}$ is p.c. if given $\mathbf{x} \in \mathbb{N}^n$ where $f(\mathbf{x}) = q$ we can compute some (a, b) from $\mathbf{x}$ such that $q = \frac{a}{b}$.

We now extend computability to the real numbers.

Definition 4.13.
1. A real number $r \in \mathbb{R}$ is said to be *computable* if there exists a computable function $f : \mathbb{N} \to \mathbb{Q}$ such that $|r - f(n)| \leq 2^{-n}$ for all $n \geq 0$.
2. A real number r is said to be a *left computably enumerable* or a left c.e. real if there is some non-decreasing sequence $\{r_n\}_{n \in \omega}$ of computable reals that converges to r.
3. A function $B : \mathbb{N} \to \mathbb{R}$ is said to be *computably enumerable* if $f(n)$ is a left c.e. real for all n.
4. A collection of functions $\{B_k : \mathbb{N} \to \mathbb{R}\}_{k \in \omega}$ is said to be *uniformly computably enumerable* if there is a computably enumerable function $B : \mathbb{N}^2 \to \mathbb{R}$ such that $B(x, k) = B_k(x)$ for all $k \in \mathbb{N}$.

Remark 4.14. At this point, the many uses of the word computable and c.e. may be confusing. However, it is always clear from the context which usage is meant. Moreover, the multiple usages all correspond to the informal idea that we can compute these functions with an effective algorithm.

With formal definitions for betting strategies and luck, we can put them together to define randomness.

Definition 4.15. A string X is *martingale-random* if no computably enumerable martingale succeeds on X. In other words, $X \notin \operatorname{Succ} B$ for any c.e. martingale B.

5 The Equivalence of our Approaches to Randomness

So far, we have characterized randomness using prefix-free complexity, Martin-Löf tests, and martingales. The first roughly said that random sequences are those that cannot be produced by a computer program using less information than contained by the whole sequence. The second gave an abstract statistical framework determining properties a random sequence does not have. Martin-Löf tests seem to say nothing about programs and the amount of contained information. Third, the martingale method says no unlucky but clever gambler should eventually break the bank on a random sequence. But betting strategies do not look like Turing machines or statistical tests. These three intuitions seem distinct. Nevertheless, we now will show that these definitions are equivalent. First, we will show that 1-randomness and Martin-Löf randomness are equivalent. This requires further relating complexity to the measure μ.

Lemma 5.1. If M is a prefix-free machine, then $\mu([\operatorname{dom} M]) = \sum_{\sigma \in M} 2^{-|\sigma|}$

Proof. [**?**] Let M be a prefix-free Turing machine. Then $\operatorname{dom} M$ is prefix-free. We claim that $\{[\sigma]\}_{\sigma \in \operatorname{dom} M}$ is a countable union of disjoint sets. Note $\operatorname{dom} M \subseteq 2^{<\omega}$, so $\{[\sigma]\}_{\sigma \in \operatorname{dom} M}$ is countable. Also, suppose for a contradiction that there exists $\pi \in [\sigma] \cap [\tau]$ for some $\sigma, \tau \in \operatorname{dom} M$. Then $\sigma \preceq \pi$ and $\tau \preceq \pi$. Without loss of generality, let $|\sigma| \leq |\tau|$. But then $\sigma \preceq \tau \preceq \pi$, which is a contradiction. Hence, $[\sigma]$ and $[\tau]$ are disjoint for all $[\sigma], [\tau] \in 2^{<\omega}$. This shows that $\{[\sigma]\}_{\sigma \in \operatorname{dom} M}$ is a countable collection of disjoint sets. Therefore, by countable additivity of μ,

$$\begin{aligned} \mu([\operatorname{dom} M] &= \mu\Big(\bigcup_{M(\sigma)\downarrow} [\sigma] \Big) \\ &= \sum_{\mathrm{M}(\sigma)\downarrow} \mu([\sigma]) \\ &= \sum_{M(\sigma)\downarrow} 2^{-|\sigma|}. \end{aligned}$$

Thus, $\mu([\operatorname{dom} M]) = \sum_{\sigma \in \operatorname{dom} M} 2^{-|\sigma|}$ for all prefix-free machines M. $\dashv$

Lemma 5.2. For a prefix-free Turing machine M and some $k \in \mathbb{N}$, set

$$S = \{\sigma : K_M(\sigma) \leq |\sigma| - k\}.$$

Then $\mu([S]) \leq 2^{-k} \mu([\operatorname{dom} M])$.

Proof. [**?**] Let $\sigma \in S$ be arbitrary. Note $S \subseteq \operatorname{dom} M$, so S is prefix-free. Then $K_M(\sigma) \leq |\sigma| - k < \infty$ by the definition of S. So there exists some $\tau_\sigma \in 2^{<\omega}$ such that $K_M(\sigma) = \tau_\sigma$ and $|\tau_\sigma| \leq |\sigma| - k$. Consequently,

$$\begin{aligned} \mu([S]) &\leq \sum_{\sigma \in S} 2^{-|\sigma|} \text{ by countable additivity and the fact that } S \text{ is prefix-free} \\ &\leq \sum_{\sigma \in S} 2^{-(|\tau_\sigma| + k)} \text{ by choice of } \tau_\sigma \\ &= 2^{-k} \sum_{\sigma \in S} 2^{-|\tau_\sigma|} \\ &\leq 2^{-k} \sum_{M(\tau)\downarrow} 2^{-|\tau|} \text{ since } S \subseteq \operatorname{dom} M \\ &= 2^{-k} \mu([\operatorname{dom} M]) \text{ by lemma 5.1.} \end{aligned}$$

Thus, $\mu([S]) \leq 2^{-k}\mu([\operatorname{dom} M])$. $\dashv$

Theorem 5.3. A sequence $A \in 2^\omega$ is 1-random if and only if it is Martin-Löf random.

Proof. [**?**] ($\Rightarrow$) Let $A \in 2^\omega$ be a Martin-Löf random sequence. We want to show that $\{U_k\}_{k\in\mathbb{N}}$ defined by $U_k = \{X : \exists n K(X \upharpoonright n) \leq n - k\}$ is a Martin-Löf test. This requires showing that $\{U_k\}$ meets two conditions.

First, we must show that $\{U_k\}$ is uniformly Σ^0_1, which is equivalent by 3.6 to showing that $\{U_k\}$ is a uniformly c.e. sequence of sets. Let $k \in \mathbb{N}$ be arbitrary. We can enumerate U_k as follows. Let $\{\sigma_i\}_{i\in\omega}$ be an enumeration of $2^{<\omega}$. For each σ_i, note that there are only finitely many strings τ with $|\tau| \leq |\sigma_i| - k$. So for each such τ, we can effectively check if $\mathcal{U}(\tau) = \sigma_i$. Hence, we can effectively check if $K(\sigma_i) \leq |\sigma_i| - k$. If it is, then include σ_i in P and repeat the procedure for σ_{i+1}. Otherwise, do not include $[\sigma_i]$ in P and repeat for σ_{i+1}. Given $X \in 2^\omega$, we can effectively check whether $\sigma_i \prec X$ for each $\sigma_i \in P$. If $K(X \upharpoonright n) \leq n - k$ for some n, then for some i we will have $\sigma_i \preceq (X \upharpoonright n)$, so this process will halt. Hence, $\{U_k\}$ is uniformly c.e. and thus uniformly Σ^0_1.

Second, note $\operatorname{dom}([\mathcal{U}]) \leq 1$. Hence, lemma 5.2 implies $\mu(U_k) \leq 2^{-k}\mu([\operatorname{dom}\mathcal{U}]) \leq 2^{-k}$. Thus, $\{U_k\}$ is a Martin-Löf test.

Since A is Martin-Löf random,

$$A \notin \bigcap_{k\in\mathbb{N}} U_k.$$

Consequently, there exists some $k \in \mathbb{N}$ such that $A \notin U_k$. But this entails that for all $n \in \mathbb{N}$ we have $K(A \upharpoonright n) > n - k$. Therefore, A is 1-random.

($\Leftarrow$) We will prove the contrapositive. Let $A \in 2^\omega$ not be a Martin-Löf random sequence. Then there exists some Martin-Löf test $\{U_n\}_{n\in\omega}$ such that $A \in \bigcap_{n\in\omega} U_n$. Since each U_n is Σ^0_1, each U_n is also uniformly computably enumerable. Furthermore, for each n the following shows that we can effectively define a prefix-free set $R_n \subseteq 2^{<\omega}$ such that $[R_n] = U_n$. By the definition of a Martin-Löf test, $\mu(U_n) \leq 2^{-n} = \mu([\tau])$ whenever $|\tau| = n$. As μ is monotone, we only need to check whether we have $[\sigma] \subseteq U_n$ when $|\sigma| \leq n$. There are only finitely many such σ, so we can effectively order them with non-decreasing length. Starting with the shortest string, include in R_n all strings σ such that $[\sigma] \subseteq R_n$ where σ does not extend a string already in R_n. With this procedure, we can define uniformly c.e. sets R_n such that $U_n = [R_n]$ for all $n \in \mathbb{N}$.

Next, let $\mathcal{X} = \bigcup_{n\geq 2} R_{n^2}$ and define $F : \mathcal{X} \to \mathbb{N}$ by $F(\sigma) = |\sigma| - n$. We claim that F is an information content measure. First, note

$$\begin{aligned}
\sum_{\sigma\in\operatorname{dom} F} 2^{-F(\sigma)} &= \sum_{\sigma\in\mathcal{X}} 2^{-(|\sigma|-n)} \text{ by the definition of } F \\
&= \sum_{n\geq 2}\sum_{\sigma\in R_{n^2}} 2^{-(|\sigma|-n)} \text{by construction of } \mathcal{X} \\
&= \sum_{n\geq 2} 2^n \sum_{\sigma\in R_{n^2}} 2^{-|\sigma|} \\
&= \sum_{n\geq 2} 2^n \mu(U_{n^2}) \text{ since } R_{n^2} \text{ is prefix-free.} \\
&\leq \sum_{n\geq 2} 2^n 2^{-n^2} \text{ since } \{U_n\} \text{ is a Martin-Löf test} \\
&= \sum_{n\geq 2} 2^{n-n^2} < \sum_{m\geq 2} 2^{-m} \text{ since } 0 > -n > n - n^2 \text{ for } n \geq 2 \\
&< 1.
\end{aligned}$$

Second, consider $\mathcal{F} = \{(\sigma, k) : k \in \mathbb{N} \text{ and } F(\sigma) \leq k\}$. We can enumerate $\mathcal{F}$ as follows. Note $(\sigma, k) \in \mathcal{F} \Leftrightarrow F(\sigma) = |\sigma| - n \leq k \Leftrightarrow |\sigma| \leq k + n$.

Step 1: There are only finitely many $\sigma \in R_{1^2}$. We can effectively order these strings by length. So for each $k \in \mathbb{N}$, we can effectively check for each $\sigma \in R_1$ whether $|\sigma| - 1 \leq k$. For all and only the strings σ that do satisfy the inequality, enumerate (σ, k) into $\mathcal{F}$. Continue until some k is reached such that $(\tau, k) \in \mathcal{F}$ for all $\tau \in R_{n^2}$. Such a k must exist since R_{1^2} is finite and each string has finite length.

Step n: Effectively enumerate R_{n^2} in the same way as R_{1^2}. Then for each (σ, k) already enumerated in $\mathcal{F}$ with $k \leq n^2$, enumerate (σ, m) into $\mathcal{F}$ for all m with $k < m \leq n^2$.

Eventually, each element of $\mathcal{F}$ will be enumerated, so $\mathcal{F}$ is c.e. Thus, F is an information content measure.

So by theorem 2.15 that K is minimal as an i.c.m., there is a constant $c \in \mathbb{N}$ such that for all $\sigma \in \mathcal{X}$ we have $K(\sigma) \leq F(\sigma) + c$. Recall $A \in \bigcap_{n\in\omega} U_n \subseteq \bigcap_{n\in\omega} U_{n^2}$. Hence, for each n there exists a $k \in \mathbb{N}$ such that $A \upharpoonright k \in U_{n^2}$, which implies $K(A \upharpoonright k) \leq F(A \upharpoonright k) + c = k - n + c$. Note for any constant $a \in \mathbb{N}$ there exists $n \in \mathbb{N}$ such that $-n + c < -a$. For the corresponding k, we have

$$K(\sigma \upharpoonright k) \leq k - n + c < k - a.$$

Therefore, A is not 1-random. $\dashv$

This theorem establishes that our machine centered approach coincides with our measure centered one. We will now show that the measure centered approach coincides with the martingale one. As the above and next proof will illustrate, the measure μ is useful for relating martingales and complexity. It would be more difficult to proof directly that the complexity approach coincides with the martingale one. Fortunately, that result follows from theorems 5.3 and 5.5.

Lemma 5.4. 1. If B is a c.e. martingale, then there exists some Martin-Löf test $\{U_n\}_{n\in\omega}$ such that $X \in \operatorname{Succ} B \Leftrightarrow X \in \bigcap_{n\in\omega} U_n$.

2. If $\{U_n\}_{n\in\omega}$ is a Martin-Löf test, then there exists some c.e. martingale B such that $X \in \bigcap_{n\in\omega} \Leftrightarrow X \in \operatorname{Succ}(B)$.

Proof. [**?**] For the first part, let B be any c.e. martingale. Without loss of generality, we can assume $B(\lambda) = 1$ since otherwise we could just divide everything by $B(\lambda)$. For each $n \in \omega$, set

$$U_n = [\{\sigma : B(\sigma) \geq 2^n\}].$$

To see that these sets are uniformly Σ^0_1, let $n \in \omega$ be arbitrary. Note that $B(\sigma)$ is a left c.e. real; i.e., some non-decreasing sequence $\{r_k\}_{k\in\mathbb{N}}$ of computable reals converges to $B(\sigma)$. Also, for each r_k there is a computable function $f_k : \mathbb{N} \to \mathbb{Q}$ such that $|r_k - f_k(m)| \leq 2^{-k}$ for all $m \in \mathbb{N}$. So we have $B(\sigma) \in U_n \Leftrightarrow \exists k[|r_k - f_k(1)| \geq 2^n]$. Thus, the sequence $\{U_n\}_{n\in\omega}$ is uniformly Σ^0_1.

Second, we must show that for each n we have $\mu(U_n) \leq 2^n$. By part (2) of Kolmogorov's inequality, $\mu(U_n) \leq \frac{B(\lambda)}{2^n} = 2^{-n}$. Thus, $\{U_n\}_{n\in\omega}$ is a Martin-Löf test. Observe

$$\begin{aligned}
&X \in \operatorname{Succ} B \\
&\Leftrightarrow \limsup_{k\to\infty} B(X \upharpoonright k) = \infty \\
&\Leftrightarrow \text{for all } n \in \omega \text{ there exists } k \in \omega \text{ such that } B(X \upharpoonright k) \geq 2^n \\
&\Leftrightarrow \text{for all n there exists k such that } [(X \upharpoonright k)] \in U_n \\
&\Leftrightarrow X \in U_n \text{ for all } n \text{ since } X \in [X \upharpoonright k] \text{ for all } k \in \omega \\
&\Leftrightarrow X \in \bigcap_{n\in\omega} U_n.
\end{aligned}$$

For the second part, let $\{U_n\}_{n\in\omega}$ be any Martin-Löf test. Take any $X \in \bigcap_{n\in\omega} U_n$. As in the proof of 5.3, let $R_0, R_1, \ldots$ be a uniformly c.e. sequence of prefix-free sets such that $U_n = [R_n]$ for all $n \in \omega$. For each $n \in \omega$, define a function $B_n : 2^{<\omega} \to \overline{\mathbb{R}}_+$ where for each $\sigma \in R_n$ we:

- add 1 to $B_n(\tau)$ for each string τ such that $\sigma \preceq \tau$, and
- add $2^{k-|\sigma|}$ to $B_n(\sigma \upharpoonright k)$ for each $k < \sigma$.

First, let's show that B_n is well-defined. It is clear that $B_n(\tau)$ is unique for all $\tau \in 2^{<\omega}$ if $B_n(\tau) \in \overline{\mathbb{R}}_+$. So we must only check for each $\tau \in 2^{<\omega}$ that $B_n(\tau) < \infty$. If $\sigma, \sigma' \in R_n$ are such that $\sigma \preceq \tau$ and $\sigma' \preceq \tau$ then either $\sigma \preceq \sigma'$ or vice versa. So there is at most one string in R_n that τ extends. Consequently, the first part of our definition of B_n adds at most 1 to $B_n(\tau)$. Hence,

$$\begin{aligned}
B_n(\tau) &\leq 1 + \sum_{\sigma\in R_n} 2^{|\tau|-|\sigma|} \\
&= 1 + 2^{|\tau|} \sum_{\sigma\in R_n} 2^{-|\sigma|} \\
&= 1 + 2^{|\tau|}\mu([R_n]) \text{ by countable additivity and the fact that } R_n \text{ is prefix-free} \\
&\leq 1 + 2^{|\tau|} \text{ since } \mu([R_n]) \leq 1 \\
&< \infty
\end{aligned}$$

Thus, B_n is well-defined.

Second, let's show that B_n is a c.e. function; i.e., let's show that $B_n(\sigma)$ is a left c.e. real for all $\sigma \in 2^{<\omega}$. Take any $\tau \in 2^{<\omega}$. We will define a non-decreasing sequence $\{x_m\}_{m\in\omega}$ of computable reals converging to $B(\tau)$. Since the collection $\{R_n\}_{n\in\omega}$ is uniformly c.e., we can enumerate R_n from n by $\sigma_0, \sigma_1, \ldots$. We will make use of the Turing thesis.

To define x_0: check if $\sigma_0 \upharpoonright k = \tau$ for any $k < |\sigma_0|$. Note that this process is effective. If there exists such a k, then set $x_0 = 2^{k-|\sigma_0|}$. Since $|\tau| = k < |\sigma_0|$, we know τ does not extend σ_0. If no such k exists, check whether $\sigma_0 \preceq \tau$. If it is, then set $x_0 = 1$. Otherwise, set $x_0 = 0$.

To define x_l assuming all x_k with $k < l$ have been defined: set $r_k = \sum_{i=0}^{k} x_i$. Using the same process as in case x_0 except with σ_k rather than σ_0, define x'_l to be either $2^{k-|\sigma_k|}$ for some $k \leq |\sigma_k|$,1, or 0. Set $x_l = r_k + x'_l$. By the definition of B_n, we know that $\{x_m\}_{m\in\omega}$ converges to $B(\tau)$.

Third, let's show that B_n is a martingale. Take any $\tau \in 2^{<\omega}$. We will show that the martingale condition holds for any possible addition given in the definition of B_n. For $\sigma \in R_n$ with $\sigma \preceq \tau$, suppose $\sigma \preceq \tau 0$ and $\sigma \preceq \tau 1$. Then we add 1 to $B_n(\tau)$, add 1 to $B_n(\tau 0)$, and also add 1 to $B_n(\tau 1)$, so the martingale condition is satisfied.

Otherwise, $\sigma \in R_n$ is such that $\sigma \upharpoonright |\tau| = \tau$ where $|\tau| < |\sigma|$. There are two cases. On the one hand, suppose $|\sigma| = |\tau| + 1$. This holds if and only if either $\sigma = \tau 0$ or $\sigma = \tau 1$. Without loss of generality, suppose $\sigma = \tau 0$. Then we add $2^{|\tau|-|\sigma|} = 2^{-1}$ to $B_n(\tau)$, 1 to $B_n(\tau 0)$, and 0 to $B_n(\tau 1)$. Since $2 \cdot 2^{-1} = 1$, the desired equality for a martingale holds. On the other hand, we could have $|\sigma| > |\tau| + 1$. Then $\sigma \upharpoonright |\tau 0| = \tau 0$ or likewise for $\tau 1$ (but not both) with $|\tau 0|, |\tau 1| < |\sigma|$. Assume without loss of generality that $\tau 0 = \sigma \upharpoonright |\tau 0|$. Consequently, we add $2^{|\tau|-|\sigma|}$ to $B_n(\sigma)$, $2^{|\tau 0|-|\sigma|} = 2 \cdot 2^{|\tau|-|\sigma|}$ to $B_n(\tau 0)$, and 0 to $B_n(\tau 0)$. In this case, the martingale condition also holds. Note that $B_n(\tau)$ is the result of all of these additions, so B_n is a martingale.

We now want to add together these martingales B_n using proposition 4.10, but this requires showing that $\sum_{n\in\omega} B_n(\lambda) < \infty$. For any $n \in \omega$, note $\mu([R_n]) \leq 2^{-n}$ since $\{[R_n]\}_{n\in\omega} = \{U_n\}_{n\in\omega}$ is a Martin-Löf test. Hence, at most only $\lambda \in R_0$, which implies $B_0(\lambda) = 1 = 2^{-0}$. For all $n > 0$, λ

does not extend any strings in R_n. Hence,

$$B_n(\lambda) \leq \sum_{\sigma \in R_n} 2^{|\lambda|-|\sigma|} = \sum_{\sigma \in R_n} 2^{-|\sigma|} = \mu([R_n]) \leq 2^{-n}.$$

Consequently, $\sum_{n\in\omega} B_n(\lambda) \leq \sum_{n\in\omega} 2^{-n} = 2$. Thus, we can define a c.e. martingale $B = \sum_{n\in\omega} B_n$.

Finally, observe

$$\begin{aligned}
X \in \bigcap_{n\in\omega} U_n &\Leftrightarrow X \in U_n \text{ for all } n \in \omega \\
&\Leftrightarrow X \in [R_n] \text{ for all } n \in \omega \\
&\Leftrightarrow \text{for all } n \text{ there exists } \sigma_n \in R_n \text{ such that } \sigma_n \prec X \\
&\Leftrightarrow \text{ for all } n \in \omega \text{ and } c \in \omega \text{ there exists } \sigma_n \in R_n \text{ s.t. either} \\
&\quad B_n(X \upharpoonright c) \geq 2^{c-|\sigma_n|} \text{ where } c < |\sigma_n| \text{ or } B_n(X \upharpoonright c) = 1 \\
&\Leftrightarrow \text{ there exists } \sigma_n \in R_n \text{ s.t. } B_n(X \upharpoonright c) = 1 \text{ for all } c \geq |\sigma_n| \\
&\Leftrightarrow \text{for all } k \in \omega \text{ we have } B(X \upharpoonright c) = \sum_n B(X \upharpoonright c) \geq k \\
&\quad \text{for any } c \geq \max\{|\sigma_0|, \ldots, |\sigma_k|\} \text{ where } X \text{ extends each } \sigma_i \in R_i \\
&\Leftrightarrow \limsup_c B(X \upharpoonright c) = \infty \text{ since } \limsup_k |\sigma_k| = \infty \text{ for } \sigma_k \in R_k \text{ as } \mu([R_k]) \leq 2^{-k} \\
&\Leftrightarrow X \in \mathrm{Succ}(B).
\end{aligned}$$

Thus, X is effectively null if and only if $X \in \mathrm{Succ}\, B$. $\dashv$

Theorem 5.5. A sequence $X \in 2^\omega$ is Martin-Löf random if and only if X is martingale-random.

Proof. [**?**] ($\Rightarrow$) Suppose X is not martingale-random. Then there exists a c.e. martingale B with $X \in \mathrm{Succ}\, B$. So by lemma 5.4, there exists a Martin-Löf test $\{U_n\}_{n\in\omega}$ with $X \in \bigcap_{n\in\omega} U_n$. Thus, X is not Martin-Löf random.

($\Leftarrow$) Suppose X is not Martin-Löf random. Then there exists a Martin-Löf test $\{U_n\}_{n\in\omega}$ with $X \in \bigcap_{n\in\omega} U_n$. By lemma 5.4 there exists a c.e. martingale B with $X \in \mathrm{Succ}\, B$. Therefore, X is not martingale-random. $\dashv$

From theorem 5.3 and 5.5, we know that 1-randomness, Martin-Löf randomness, and martingale-randomness are all equivalent. In the literature, this equivalent notion is usually just referred to as 1-randomness. The convergence of these three approaches offers substantial evidence that 1-randomness is *the* formal definition of the randomness. Of course, convergence alone is not satisfactory evidence, and there are other approaches that are not equivalent to those presented here. We do not have the space explore them here, but many of the tools we have seen will be useful in understanding them. The theory of computation has taken us a long way to rigorously understanding randomness. It can take us much further.

References

[1] Conrad, Keith. *Infinite Series in p-adic Fields*. 2016. Web

[2] Cooper, Barry S. *Computability Theory*. Boca Raton: Chapman & Hall/CRC, 2004. Print.

[3] Downey, Rodney G. and Denis R. Hirschfeldt. *Algorithmic Randomness and Complexity*. New York: Springer, 2010. Print.

[4] Enderton, Herbert B. *A Mathematical Introduction to Logic*. Harcourt/Academic Press, 2001. Print.

[5] Nies, André. *Computability and Randomness*. Oxford: Oxford University Press, 2009. Print.

[6] Simpson, Stephen G. *Topics in Logic and Foundations: Spring 2005. Math 497A: Computability, Unsolvability, Randomness*. Pennsylvania State University, 1 Nov. 2005. Web.